11/24 11/24 CP CLUSTER DIAG AH-F608A-MC
CJKDEA0 FICHE 1 OF 3
AUG 1981
COPYRIGHT© 1981
MADE IN USA

11/24 11/24 CP CLUSTER DIAG AH-F608A-MC AUG 1981
CJKDEA0 FICHE 2 OF 3 COPYRIGHT© 1981
MADE IN USA

11/24 11/24 CP CLUSTER DIAG AH-F608A-MC
CJKDEA0 FICHE 3 OF 3
AUG 1981
COPYRIGHT© 1981
MADE IN USA
digital

B 1

.REM_

IDENTIFICATION

PRODUCT CODE: AC-F605A-MC

PRODUCT NAME: CJKDEA0 11/24 CP CLUSTER DIAG

PRODUCT DATE: MAY-81

MAINTAINER: DIAGNOSTIC ENGINEERING

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C): 1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	DEC/X11

C 1

HISTORY

REVISION A		FIRST RELEASE OF DIAGNOSTIC

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION
1.1 ABSTRACT
1.2 SYSTEM REQUIREMENTS
1.3 RELATED DOCUMENTS AND STANDARDS
1.4 DIAGNOSTIC HIERARCHY PREREQUISITES
1.5 ASSUMPTIONS

2.0 OPERATING INSTRUCTIONS
2.1 LOADING AND STARTING PROCEDURE
2.2 PROGRAM OPTIONS
2.3 EXECUTION TIMES

3.0 ERROR INFORMATION
3.1 ERROR REPORTING PROCEDURES
3.2 ERROR HALTS

4.0 PERFORMANCE AND PROGESS REPORTS
4.1 PERFORMANCE REPORTS
4.2 PROGRESS REPORTS

5.0 DEVICE INFORMATION TABLES

6.0 PROGRAM DESCRIPTION
6.1 PROGRAM EXECUTION CHARACTERISTICS
6.2 SUBTEST SUMMARIES
6.3 SPECIAL SUBROUTINE DESCRIPTION

7.0 LISTING

1.0 ABSTRACT

THIS PROGRAM IS A GO-NOGO TEST FOR THE 11/24 CPU BOARD. IT TESTS THE CPU INCLUDING EIS, THE MMU, THE FPP, THE LTC AND BOTH SLU'S. IT DOES NOT CONTAIN THE CAPABILITIES OF SCOPE LOOPING, ERROR RECOVERY OR PRINTING OF ERROR INFORMATION. ERROR HALTS DO INDICATE WHICH DEVICE FAILED TO ALLOW THE TECHNICIAN TO DETERMINE WHICH DIAGNOSTIC TO USE TO FIX THE BOARD OR WHAT FIELD REPLACEABLE UNIT (FRU) MAY FIX THE BOARD. THE PROGRAM WILL RUN UNDER THE ACT AND APT MANUFACTURING SYSTEMS AND IS CHAINABLE UNDER XXDP.

1.1 SYSTEM REQUIREMENTS

A. HARDWARE REQUIREMENTS

- CPU BASED ON DCF11-AA CHIP SET WITH KTF11-AA AND KEF11-AA CHIPS PRESENT.
- LTC
- 2 SLU'S COMPATIBLE WITH 11/24 SLU SPECIFICATION.
- 28K OF MEMORY
- THE SECOND SLU MUST HAVE TURN AROUND CONNECTOR.

B. SOFTWARE ENVIRONMENTS

- APT (MULTI-CPU TESTER)
- ACT
- XXDP (SLIDE)
- STAND-ALONE

1.2 RELATED DOCUMENTS AND STANDARDS

- ASSEMBLED WITH SYSMAC; SEE FIRST PAGE OF LISTING FOR REVISION NUMBER.
- M7133 MODULE SPECIFICATION
- DIAGNOSTIC ENGINEERING FUNCTIONAL SPECIFICATION FOR SPECIAL MANUFACTURING TEST BGI-79-003-00-U.
- DIAGNOSTIC ENGINEERING PROJECT PLAN FOR 11/24) BGI-78-002-02-U.

- DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS
175-003-009-02.

1.3 PREREQUISITE DIAGNOSTICS

NONE

1.4 ASSUMPTIONS

THIS PROGRAM ASSUMES THE MACHINE IS UP SUFFICIENTLY TO ALLOW PROPER OPERATION OF THE MICRO-ODT OF THE DCF11-AA CHIP SET.

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES

TO LOAD AND START THIS PROGRAM USE THE STANDARD PROCEDURES FOR THE DIAGNOSTIC SOFTWARE ENVIRONMENT THAT IS BEING USED.

2.2 PROGRAM OPTIONS

THIS PROGRAM USES THE SOFTWARE SWITCH LOCATION 176 IF PROGRAM IS NOT BEING RUN UNDER APT MODE (BIT 0 SET OF LOCATION $ENV). IF PROGRAM IS BEING RUN IN APT MODE THE LOCATION $SWREG IN THE APT ETABLE IS USED TO STORE OPERATING SWITCHES.

BIT #	DEFINITION
15-6	NOT USED
5	0 - PROGRAM RESERVED -- PROGRAM WILL CLEAR IF CIS CHIP SET NOT ON BOARD
	1 - PROGRAM RESERVED -- PROGRAM WILL SET IF CIS CHIP SET IS ON BOARD
4	0 - TEST SLU2 OF 11/24
	1 - INHIBIT TESTING OF SLU2
3	0 - TEST LTC OF 11/24
	1 - INHIBIT TESTING OF LTC
2	0 - TEST SLU1 OF 11/24
	1 - INHIBIT TESTING OF SLU1
1	0 - TEST FPP INSTRUCTION SET
	1 - INHIBIT TESTING OF FPP
0	0 - TEST MEMORY MANAGEMENT UNIT
	1 - INHIBIT TESTING OF MMEMORY MANAGEMENT UNIT

2.3 EXECUTION TIMES

FIRST PASS RUNTIME (WORST CASE).....45 SEC
LONGEST TEST TIME.....30 SEC
ADDITIONAL RUNTIME (EXTRA UNITS).....NONE
LONGEST PASS TIME.....45 SEC

3.0 ERROR INFORMATION

3.1 ERROR REPORTING PROCEDURES

THE PROGRAM DOES NOT TYPE OUT ANY ERROR REPORTS OF ITS OWN BUT TAKES ADVANTAGE OF THE HARDWARE FEATURE THAT TYPES THE PC WHEN A HALT OCCURS. WHEN AN ERROR IS DETECTED THE PROGRAM JUMPS TO ONE OF SEVEN HALT ROUTINES. THE ROUTINES SIMPLY MOVE A FATAL ERROR NUMBER INTO LOCATION $FATAL, SET THE FATAL ERROR FLAG IN LOCATION $MSGTY AND EITHER HALT OR IF ON APT DO A BRANCH DOT. THE OPERATOR HAS THREE WAYS TO DETERMINE THE FAILING DEVICE; 1) BY EXAMINING LOCATION $FATAL, 2) BY DETERMINING THE HALT ADDRESS AND LOOKING UP THE ADDRESS IN THE LISTING AND 3) BY EXAMINING LOCATION $TESTN WHICH WILL CONTAIN THE TEST NUMBER BEING EXECUTED.

3.2 ERROR HALTS

FOR DISCUSSION SEE SECTION 3.1. THE LABELS FOR THE HALTS AND THE DEVICE THEY INDICATE HAVING FAILED ARE:

CPUHLT: CPU
MMUHLT: MMU
FPPHLT: FPP
LTCHLT: LTC
SL1HLT: SLU1
SL2HLT: SLU2
COMHLT: SYSTEM INTERACTION

UPON RECEIVING THE ERROR HALT ADDRESS (PC) FROM THE MICRO-ODT THE OPERATOR CAN LOOK UP THESE TAGS IN THE SYMBOL TABLE AT THE END OF THE LISTING TO DETERMINE WHICH HALT WAS EXECUTED. NOTE: THE PC SUPPLIED BY THE MICRO-ODT WILL BE THE HALT ADDRESS PLUS 2.

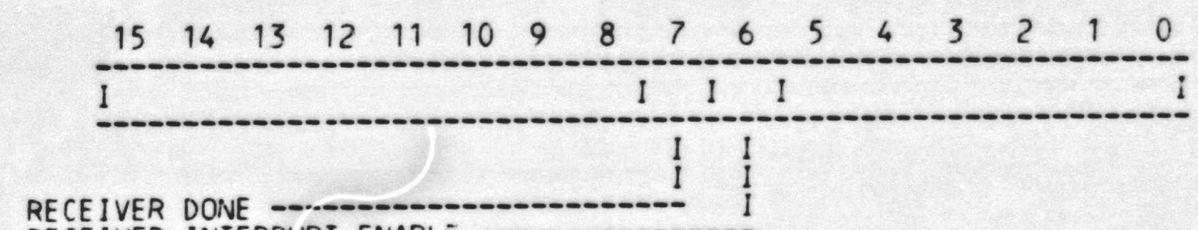
THE OPERATOR CAN DETERMINE WHICH TEST THE ERROR OCCURRED IN BY EXAMINING LOCATION "$TESTN". THE TEST NUMBERS EQUATE TO FAILING DEVICES AS FOLLOWS:

DEVICE	CONTENTS OF $TESTN
CPU	000001
MMU	000002

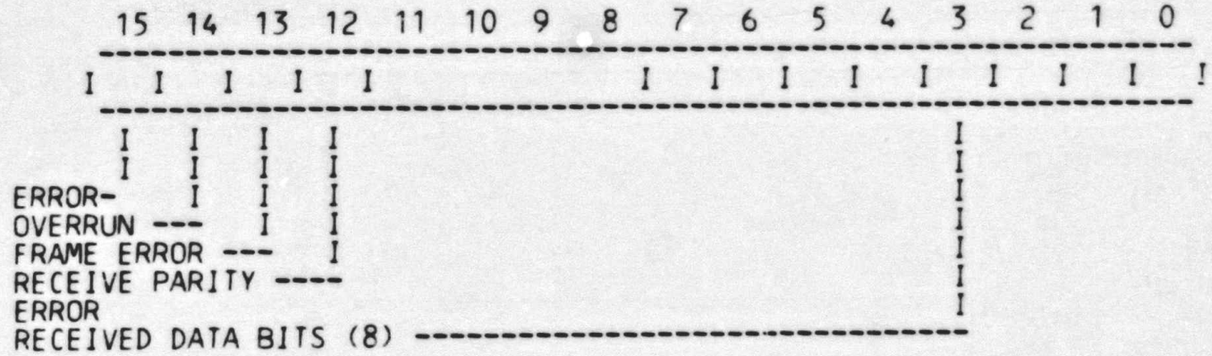
```
                                                    H 1
                        FPP             000003
                        SLU1            000004
                        LTC             000005
                        SLU2            000006
                        SYSTEM
                        INTERACTION     000007


4.0  PERFORMANCE AND PROGRESS REPORTS

     THE ONLY REPORT TYPED BY THIS PROGRAM IS  THE  END  PASS  MESSAGE
     WHICH IS:

               CJKDEA END OF PASS #XXX

     WHERE XXX IS THE DECIMAL NUMBER OF PASSES COMPLETED.
```

5.0 DEVICE INFORMATION TABLES

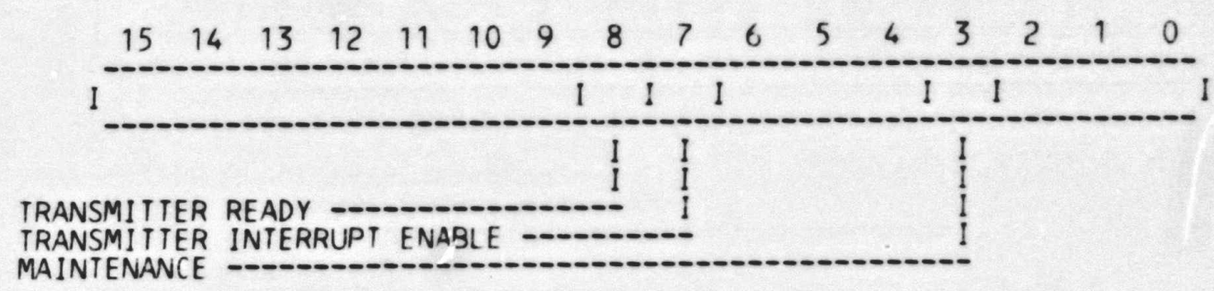
SLU1 RCSR



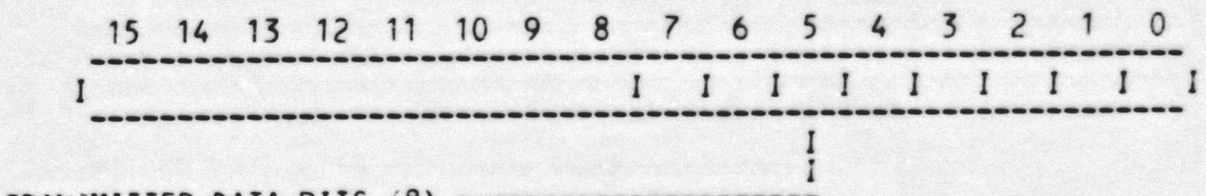
SLU1 RBUF



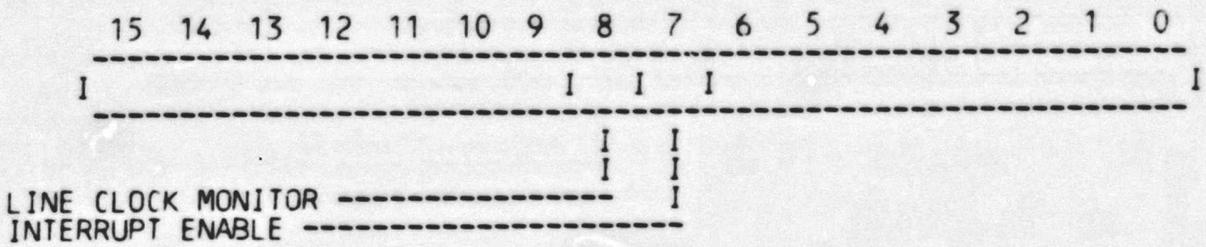
SLU1 XCSR



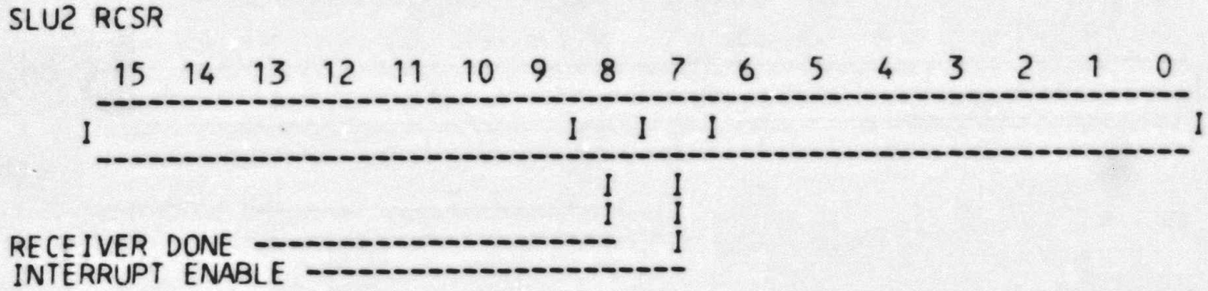
SLU1 XBUF



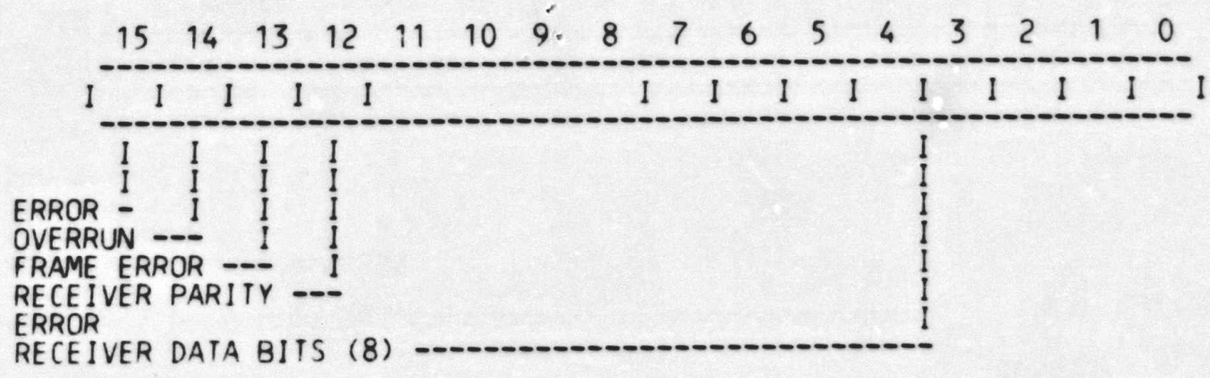
LTC CSR



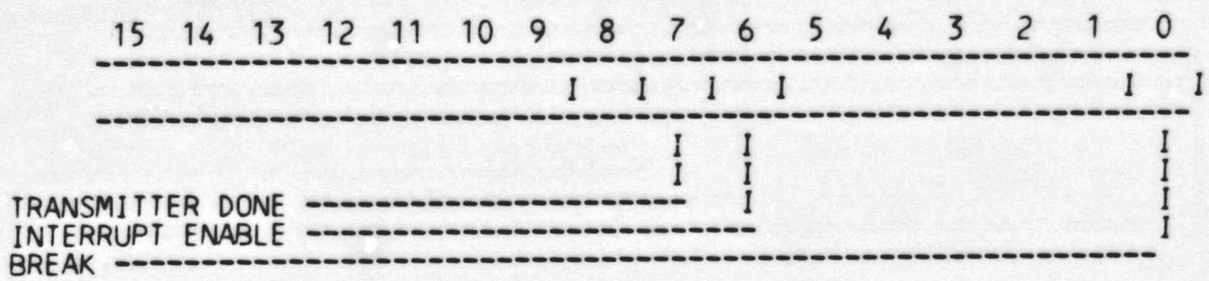
SLU2 RCSR



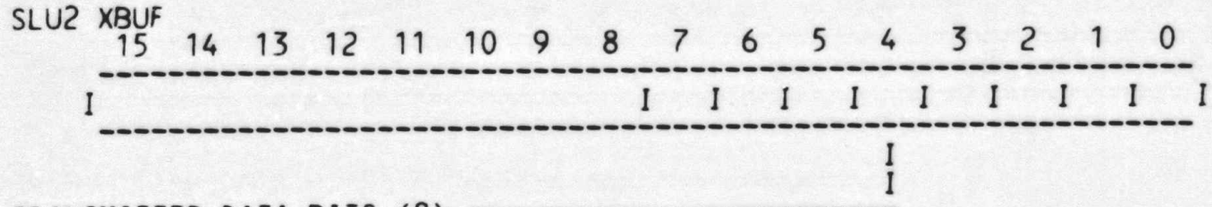
SLU2 RBUF



SLU2 XCSR



SLU2 XBUF



6.0 PROGRAM DESCRIPTION

6.1 PROGRAM EXECUTION CHARACTERISTICS

THIS PROGRAM RUNS THE SAME UNDER ALL DIAGNOSTIC MONITORS. WHEN THE TEST IS STARTED AT ADDRESS 200 OCTAL THE TESTING IS DONE AND ON COMPLETION THE TITLE IS TYPED AS PART OF THE END OF PASS MESSAGE.

6.2 SUB-TEST SUMMARIES

6.2.1 CENTRAL PROCESSING UNIT SUBTEST -

THESE TESTS CHECK THE BASIC INSTRUCTION SET AND ADDRESSING MODES, THE EXTENDED ELEVEN INSTRUCTION SET (EIS) AND TRAPS TESTING. IT IS EQUIVALENT TO CJKDB.

6.2.2 MEMORY MANAGEMENT UNIT SUBTEST -

THESE TESTS ARE THE SAME AS IN CJKDA, THE KEF11-AA TEST. THE PROGRAM BEGINS BY TESTING SOME OF THE INTERNAL CPU DATA AND ADDRESS PATHS AND ADDRESS DETECTION LOGIC. NEXT THE MEMORY MANAGEMENT REGISTERS ARE CHECKED FOR DATA RELIABILITY, THEN RELOCATION CAPABILITIES (FORMATION OF A PHYSICAL ADDRESS FROM A VIRTUAL ADDRESS AND ASSOCIATED PAGE DESCRIBTOR (PDR) INFORMATION). FINALLY THE ABORT AND STATUS SEGMENTS OF THE LOGIC ARE CHECKED.

6.2.3 FLOATING POINT PROCESSOR SUBTEST -

THE FLOATING POINT PROCESSOR SUBTEST CHECKS FLOATING POINT REGISTERS FIRST USING A LIMITED NUMBER OF FLOATING POINT INSTRUCTIONS. IT THEN VERIFIES THE REST OF THE FLOATING POINT INSTRUCTION SET USING A NUMBER OF DATA PATTERNS FOR EACH INSTRUCTION.

6.2.4 SERIAL LINE UNIT (SLU1) SUBTEST -

THESE TEST FIRST CHECK THE SLU'S REGISTERS FOR ADDRESSING AND DATA CAPABILITIES. THEN USING THE MAINTENANCE MODE OF THE SLU THEY CHECK THE DATA PATHS AND INTERRUPT LOGIC.

6.2.5 LINE TIME CLOCK (LTC) SUBTEST -

FIRST THE REGISTER IS CHECKED FOR ADDRESSING AND BIT SETTING CAPABILITIES THEN THE INTERRUPT LOGIC IS CHECKED. THERE IS ALSO A REPEATABILITY TEST FOR THE CLOCK.

6.2.6 SERIAL LINE UNIT 2 SUBTEST -

THE TESTING DONE HERE IS THE SAME AS FOR SLU1 (SEE SECTION 6.2.4). THE ONLY DIFFERENCE IS THAT SINCE SLU2 DOES NOT HAVE A MAINTENANCE WRAP-AROUND LIKE SLU1 THE TESTING REQUIRES AN EXTERNAL JUMPER TO BE PRESENT.

6.2.7 BLAST SUBTEST

THIS TEST CHECKS THE ABILITY OF THE 11/24 TO HANDLE SYSTEM INTERACTION. THE CPU HAS TO HANDLE DEVICES AT DIFFERENT PRIORITY LEVELS AND ARBITRATE BETWEEN THEM AND ITS OWN PRIORITY. THE TEST SETS UP ALL DEVICES TO INTERRUPT THEN ENABLES THEM ALL AT ONCE. THE SLU'S TRANSFER DATA UNTIL THEY TRANSFER 400(8) BYTES OR UNTIL ONE SECOND (60 TICKS) OF THE LINE CLOCK HAS BEEN RECEIVED. THE PROGRAM THEN VERIFIES THE NUMBER OF TRANSMTTER INTERRUPTS IS EQUAL TO THE NUMBER OF RECEIVER INTERRUPTS. FINALLY THE DATA TRANSFERRED BY EACH DEVICE IS CHECKED.

6.3 SPECIAL SUBROUTINE DESCRIPTIONS

THE ONLY SPECIAL SUBROUTINES ARE THE ERROR ROUTINES EACH SUBTEST HAS IS OWN. THE ROUTINES SIMPLY SET THE FATAL ERROR FLAG IN THE APT MAILBOX AND EITHER ''BRANCH SELF'' OR ''HALT''. THIS CHOICE IS DETERMINED IN THE INITIALIZE PORTION OF THE PROGRAM AND IS A ''BRANCH SELF'' IF RUNNING UNDER APT OR A ''HALT'' IF RUNNING UNDER ANY OTHER MONITOR.

```
                                                    .TITLE  CJKDE-A  11/24 CPU CLUSTER DIAG.
                                                    .ENABLE ABS
                                                    .NLIST  CND,MC,MD
                                                    .LIST   ME
000240                                              SCOPE=NOP
000007                                              R7=%7
000006                                              R6=%6
177776                                              PS=177776
177564                                              TPS=177564
177566                                              TPB=177566
140000                                              USRM=140000
030000                                              PUSRM=30000
000006                                              SP=%6
000006                                              R6=%6
000003                                              TAB=%3
000001                                              LAST=%1
000005                                              FIRST=%5
000002                                              R2=%2
000000                                              HLT=HALT
000003                                              TRT=3
000004                                              ITRAP5=4
000004                                              RTRAP5=4            ;RESERVED INST AND ILLEGAL ADDRESSES
000014                                              RTRAP4=14           ;FOR TRACE TRAP
000030                                              RTRAP3=30           ;FOR EMULATOR TRAP
000020                                              RTRAP2=20           ;FOR IOT TRAP
000034                                              RTRAP1=34           ;FOR TRAP INST
177564                                              TTCSR=177564
177560                                              TRCSR=177560
177562                                              TKB=177562
177566                                              TPB=177566
000240                                              BELL=240
000240                                              NOP=240
177776                                              STATUS=177776
000077                                              TRAPA=77
000010                                              RTRAP=10
004700                                              ILLA=004700
000100                                              ILLB=100
177776                                              CC=177776
001000                                              KERSTK=STBOT
000600                                              USESTK=STBOT-200
104377                                              EMTA=104377
104777                                              TRAPC=104777
000001                                              APTENV=1
                                                .SBTTL  ACT11 HOOKS

                                                ;*******************************************************************************
                                                ;HOOKS REQUIRED BY ACT11
000400                                                  $SVPC=.             ;SAVE PC
000046                                                  .=46
000046  131056                                          $ENDAD              ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                                  .=52
000052  000000                                          .WORD   0           ;;2)SET LOC.52 TO ZERO
000400                                                  .=$SVPC             ;; RESTORE PC
001000                                                  .=1000
```

```
                                .SBTTL  APT MAILBOX-ETABLE

                                ;*****************************************************************************
                                .EVEN
                                $MAIL:                          ;;APT MAILBOX
                                $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
                                $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
                                $TESTN: .WORD   ATESTN          ;;TEST NUMBER
                                $PASS:  .WORD   APASS           ;;PASS COUNT
                                $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
                                $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
                                $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
                                $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
                                $ETABLE:                        ;;APT ENVIRONMENT TABLE
                                $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
                                $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
                                $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
                                $USWR:  .WORD   AUSWR           ;;USER SWITCHES
                                $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
                                ;*                              BITS 15-11=CPU TYPE
                                ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                                      11/70=06,PDQ=07,Q=10
                                ;*                              BIT 10=REAL TIME CLOCK
                                ;*                              BIT  9=FLOATING POINT PROCESSOR
                                ;*                              BIT  8=MEMORY MANAGEMENT
001030                          $ETEND:
                                .MEXIT
                                .SBTTL  APT PARAMETER BLOCK

                                ;*****************************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;*****************************************************************************
                                        .$X=.   ;;SAVE CURRENT LOCATION
                                        .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
                                        200     ;;FOR APT START UP
                                        .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
                                        $APTHDR ;;POINT TO APT HEADER BLOCK
                                        .=.$X   ;;RESET LOCATION COUNTER
                                ;*****************************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001030                          $APTHD:
001030  000000                 $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001032  001000                 $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001034  000010                 $TSTM:  .WORD   10      ;;RUN TIM OF LONGEST TEST
001036  000025                 $PASTM: .WORD   25      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001040  000000                 $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001042  000014                         .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
                                ;*****************************************************************************
                                ;SOME POINTERS TO CPU TRAP HANDLERS
                                ;*****************************************************************************
                                        .=4
000004  021216                  TO4
000006  000000                  0
000010  021220                  TO10
```

```
                                                        0
                                                        T014
                                                        0
                                                        T020
                                                        0
                                                        .=30
                                                        T030
                                                        0
                                                        T034
                                                        0
                                                        .=114
                                                        T0114
                                                        0
                                                        .=244
                                                        T0244
                                                        0
                                                        T0250
                                                        0

                                        .=172
                                MTFLAG:         0       ;MULTI-TESTER ACTIVE BIT
                                DISPREG:        0       ;SOFTWARE DISPLAY REGISTER
                                SWREG:          0       ;SOFTWARE SWITCH REGISTER

                                ;*****************************************************************************************
                                ;DATA TABLE FOR USE IN ADDRESSING MODE TESTS
                                ;*****************************************************************************************
                                        .=370
000370  000000  000000  000000          0,0,0,0,0,0
000376  000000  000000  000000
000404  000001  000001  177777          1,1,-1
                                ;*****************************************************************************************
                                ;SET UP STARTING ADDRESS
                                .=1000
001000  000000                  STBOT:  .WORD 0                         ;STACK POINTER

                                        .=200
000200  000167  000774                  JMP     START
000204  012706  001000                  MOV     #STBOT,R6               ;SET STACK POINTER
000210  012702  001004                  MOV     #$TESTN,R2              ;SET MAILBOX POINTER
000214  000137                          JMP     @(PC)+                  ;JUMP TO SUBTEST
000216  000000                          0                               ;ADDR. OF SUBTEST GOES HERE

                                .=1200
                                .SBTTL **STARTING OF CPU TEST **
001200  012737  000000  001006  START:  MOV     #0,@#$PASS      ;CLEAR PASS COUNT
001206  012737  131130  000024  RESTRT: MOV     #PWRDN,@#24             ;SET UP FOR POWER FAIL
001214  012706  001000                  MOV     #STBOT,R6       ;SET UP STACK
001220  012737  001250  000004          MOV     #1$,@#4         ;SET UP FOR TIMEOUT IF NO MULTI TESTER
001226  012737  000340  000006          MOV     #340,@#6
001234  012737  000002  164000          MOV     #2,@#164000     ;SET BIT1 FOR MULTI TESTER
001242  012737  000001  000172          MOV     #1,@#MTFLAG     ;SET FLAG TO INDICATE MULTI-TESTER
001250  012737  021216  000004  1$:     MOV     #T04,@#4                ;SET TRAP CATCHER
001256  012737  000000  000006          MOV     #0,@#6          ;SET HALT BACK IN LOCATION 6
001264  012706  001000                  MOV     #STBOT,R6       ;INITIALIZE STACK POINTER
001270  012737  000001  001004          MOV     #1,@#$TESTN     ;SET TEST NUMBER TO 1
```

```
001276  012737  000000  001002          MOV     #0,@#$FATAL     ;CLEAR ERROR INDICATOR
001304  012737  000000  001000          MOV     #0,@#$MSGTY     ;CLEAR MESSAGE TYPE(FOR APT)
001312  105737  001020                  TSTB    @#$ENV          ;RUNNING ON APT
001316  001025                          BNE     TS1             ;IF YES DO BRANCH SELF ON ERROR
001320  012737  000000  002076          MOV     #0,@#CPUHLT     ;IF NOT THEN PUT A HALT IN ON ERROR
001326  012737  000000  050326          MOV     #0,@#MMUHLT
001334  012737  000000  120176          MOV     #0,@#FPHLT
001342  012737  000000  124100          MOV     #0,@#LTCHLT
001350  012737  000000  122670          MOV     #0,@#SL1HLT
001356  012737  000000  126670          MOV     #0,@#SL2HLT
001364  012737  000000  130006          MOV     #0,@#COMHLT

;*****************************************************************************************
;
;       THIS TEST EXECUTES EVERY POSSIBLE BRANCH WITH EVERY POSSIBLE
;CONDITION CODE COMBINATION.
;       THE ROUTINE USES TWO TABLES.  THE BRANCH TABLE HOLDS ALL THE
;POSSIBLE BRANCH INSTRUCTIONS, THE OTHER TABLE (YNTAB) HOLDS BIT MAPS FOR
;EACH BRANCH.  A ONE IN THE BIT MAP INDICATES THAT THE CORRESPONDING
;BRANCH INSTRUCTION SHOULD BRANCH FOR THE CONDITION CODE SETTING WHICH
;CORRESPONDS TO THE BIT POSITION WITHIN THE MAP.  FOR EXAMPLE IF THE LEFT
;MOST BIT IS A ONE THEN THE CORRESPONDING BRANCH INSTRUCTION SHOULD BRANCH
;WHEN THE CONDITION CODES ARE 0.
;       THE ROUTINE CONSISTS OF NESTED LOOPS; THE OUTER LOOP SETS UP
;ALL THE POSSIBLE BRANCH INSTRUCTIONS.  THE INNER LOOP SETS UP EVERY POSSIBLE
;CONDITION CODE FOR EACH BRANCH.
;       THE BIT MAP IS USED TO SET THE ADDRESS LOCATION IN TWO
;JUMP MODE 3 INSTRUCTIONS.  THE ADDRESSES ARE CHANGED TO ALLOW THE
;PROGRAM TO CONTINUE OR JUMP TO AN ERROR ROUTINE DEPENDING UPON
;WHETHER IT HANDLED THE BRANCH INSTRUCTION CORRECTLY.
;       AT ANY ERROR HALT, LOCATION, BRH, HOLDS THE BRANCH INSTRUCTION
;UNDER TEST AND LOCATION, CC, HOLDS THE VALUE OF THE CONDITION CODES
;AT THE TIME THE BRANCH WAS EXECUTED.
;
;*****************************************************************************************
;TEST 1 TEST THE BRANCH ROM
;*****************************************************************************************
001372                          TS1:
001372  012700  021122          SETUP:  MOV     #BRTAB,R0       ;INITIALIZE BRANCH TABLE POINTER
001376  012704  021160                  MOV     #YNTAB,R4       ;INITIALIZE YES/NO BRANCH MAP POINTER
001402  012767  000017  000130          MOV     #15.,BRCT       ;INITIALIZE BRANCH TABLE COUNT
001410  012067  000110          SETBR:  MOV     (R0)+,BRH       ;GET NEXT BRANCH INST.
001414  012401                          MOV     (R4)+,R1        ;GET NEXT BRANCH MAP
001416  012767  177777  000074          MOV     #-1,CC1         ;INITIALIZE CONDITION CODE VALUE
001424  012703  000020                  MOV     #16.,R3         ;INITIALIZE CONDITION CODE COUNT
001430  005267  000064          SETCC:  INC     CC1             ;SET FOR NEXT CC VALUE
001434  032701  100000                  BIT     #100000,R1      ;SEE IF SHOULD BR W/ THESE CC'S
001440  013705  177776                  MOV     @#177776,R5     ;SIMULATE A JNE
001444  042705  177773                  BIC     #177773,R5      ;       (JUMP NOT EQUAL)
001450  000165  001454                  JMP     .+4(R5)         ;       TO SET2BR
001454  000167  000020                  JMP     SET2BR
001460  012767  001542  000042          MOV     #CONT,NBR       ;SET TO CONTINUE IF NO BRANCH
001466  012767  001536  000040          MOV     #ER,YBR         ;SET TO REPORT ERROR IF BRANCH
001474  000167  000014                  JMP     AROUN           ;GO AROUND OPPOSITE CONDITION
001500  012767  001536  000022  SET2BR: MOV     #ER,NBR         ;SET TO REPORT ERROR IF NO BRANCH
001506  012767  001542  000020          MOV     #CONT,YBR       ;SET TO CONTINUE IF BRANCH
```

```
                                AROUN:  ROL     R1              ;UPDATE BIT MAP

                                        MOV     (PC)+,@(PC)+    ;SET CONDITION CODE
                                CC1:    0                       ;NEW CC VALUE GOES HERE
                                        177776
                                BRH:    0                       ;BRANCH INST. GOES HERE
                                        JMP     @(PC)+          ;THIS JUMP IF NO BRANCH
                                NBR:    0                       ;WHERE TO GO IF NO BRANCH OCCURS
                                        JMP     @(PC)+          ;THIS JUMP IF BRANCH OCCURS
                                YBR:    0                       ;WHERE TO GO IF BRANCH OCCURS
                                ER:
                                        BR      ERROR1          ;
                                BRCT:   0
                                CONT:   DEC     R3              ;CC'S DONE?
                                        MOV     @#177776,R5     ;SIMULATE A JNE
                                        BIC     #177773,R5      ;       (JUMP NOT EQUAL)
                                        JMP     .+4(R5)         ;       TO SETCC
                                        JMP     SETCC
                                        DEC     BRCT            ;BR'S DONE?
                                        MOV     @#177776,R5     ;SIMULATE A JNE
                                        BIC     #177773,R5      ;       (JUMP NOT EQUAL)
                                        JMP     .+4(R5)         ;       TO SETBR
                                        JMP     SETBR
                                        MOV     #357,R0         ;IF THIS TEST IS DONE SET UP R0 FOR THE NEXT
                                                                ;SEVEN TESTS. THIS IS SAVING 4 LOCATIONS PER
                                                                ;TEST WHICH I NEED BECAUSE BRANCHES WERE OUT
                                                                ;OF BOUNDS.

                                ;*******************************************************************************
                                ;TEST 2 TEST TRAP OF RESERVED INSTRUCTION
                                ;*******************************************************************************
                                TS2:
                                        MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
                                        MOV     #1$,RTRAP               ;SET UP NEW PC IN VECTOR
                                        MOV     R0,RTRAP+2              ;SET UP NEW PSW IN VECTOR
                                        CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
                                        TRAPA                   ;DO TRAP
                                8$:
                                        BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
                                1$:     CMP     R0,STATUS       ;IS NEW STATUS CORRECT
                                        BNE     ERROR1          ;NEW STATUS WRONG
                                2$:     CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
                                        BNE     ERROR1          ;STACK DID NOT DECREMENT CORRECTLY
                                3$:     CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
                                        BNE     ERROR1          ;PROPER PC WAS NOT SAVED
                                4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
                                        BNE     ERROR1          ;WRONG PSW SAVED
                                5$:     ADD     #4,SP           ;RESET STACK POINTER
                                        MOV     #6$,RTRAP               ;SET UP NEW PC IN VECTOR
                                        CLR     RTRAP+2         ;SET UP NEW PSW IN VECTOR
                                        MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
                                        TRAPA                   ;DO TRAP
                                        BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
                                6$:     TST     STATUS          ;IS NEW PSW CORRECT
                                        BNE     ERROR1          ;NEW PSW WRONG
```

```
                                7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
001730  020067  177042                BNE     ERROR1          ;OLD STATUS WRONG
001734  001052
                                ;*****************************************************************************
                                ;TEST 3 TEST TRAP OF TRAP INSTRUCTION
                                ;*****************************************************************************
                                TS3:
001736                                  MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
001736  012706  001000                    MOV     #1$,RTRAP1              ;SET UP NEW PC IN VECTOR
001742  012767  001764  176064            MOV     R0,RTRAP1+2             ;SET UP NEW PSW IN VECTOR
001750  010067  176062                    CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
001754  005067  176016                    TRAP                    ;DO TRAP
001760  104400
001762                          8$:
001762  000437                    BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
001764  020067  176006          1$:       CMP     R0,STATUS       ;IS NEW STATUS CORRECT
001770  001034                    BNE     ERROR1          ;NEW STATUS WRONG
001772  020627  000774          2$:       CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
001776  001031                    BNE     ERROR1          ;STACK DID NOT DECREMENT CORRECTLY
002000  022767  001762  176766  3$:       CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
002006  001025                    BNE     ERROR1          ;PROPER PC WAS NOT SAVED
002010  005767  176762          4$:       TST     STBOT-2         ;WAS OLD PSW SAVED
002014  001022                    BNE     ERROR1          ;WRONG PSW SAVED
002016  062706  000004          5$:       ADD     #4,SP           ;RESET STACK POINTER
002022  012767  002044  176004            MOV     #6$,RTRAP1              ;SET UP NEW PC IN VECTOR
002030  005067  176002                    CLR     RTRAP1+2                ;SET UP NEW PSW IN VECTOR
002034  010037  177776                    MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
002040  104777                            TRAPC                   ;DO TRAP WITH LOWER BYTE ALL ONES
002042  000407                            BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
002044  005767  175726          6$:       TST     STATUS          ;IS NEW PSW CORRECT
002050  001004                    BNE     ERROR1          ;NEW PSW WRONG
002052  020067  176720          7$:       CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
002056  001001                    BNE     ERROR1          ;OLD STATUS WRONG
002060  000407                    BR      CPUHLT+2        ;GET OVER ERROR CALL TO NEXT TEST IF NO ERROR
                                ;*****************************************************************************
                                ;THIS ERROR IS USED FOR THE ENTIRE CPU,TRAPS AND EIS PORTION OF
                                ;THIS TEST
                                ;*****************************************************************************
002062  012737  000001  001002  ERROR1: MOV     #1,@#$FATAL                     ;SET UP FATAL ERROR NUMBER
002070  012767  000001  176702          MOV     #1,$MSGTY                       ;SET FATAL ERROR FLAG
002076  000777                  CPUHLT: BR      .
                                ;*****************************************************************************
                                ;TEST 4 TEST TRAP OF IOT INSTRUCTION
                                ;*****************************************************************************
002100                          TS4:
002100  012706  001000                  MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
002104  012767  002126  175706          MOV     #1$,RTRAP2              ;SET UP NEW PC IN VECTOR
002112  010067  175704                  MOV     R0,RTRAP2+2             ;SET UP NEW PSW IN VECTOR
002116  005067  175654                  CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
002122  000004                          IOT                     ;DO TRAP
002124                          8$:
002124  000756                    BR      ERROR1          ;INSTRUCTION FAILED TO TRAP
002126  020067  175644          1$:       CMP     R0,STATUS       ;IS NEW STATUS CORRECT
002132  001353                    BNE     ERROR1          ;NEW STATUS WRONG
002134  020627  000774          2$:       CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
002140  001350                    BNE     ERROR1          ;STACK DID NOT DECREMENT CORRECTLY
002142  022767  002124  176624  3$:       CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
002150  001344                    BNE     ERROR1          ;PROPER PC WAS NOT SAVED
```

```
                                ;WAS OLD PSW SAVED
                                ;WRONG PSW SAVED
                                ;RESET STACK POINTER
                                        ;SET UP NEW PC IN VECTOR
                                        ;SET UP NEW PSW IN VECTOR
                                ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
                                ;DO TRAP
                                ;INSTRUCTION FAILED TO TRAP
                                ;IS NEW PSW CORRECT
                                ;NEW PSW WRONG
                                ;WAS OLD STATUS STORED
                                ;OLD STATUS WRONG
                                ;*****************************************************************************
                                ;TEST 5 TEST TRAP OF EMT INSTRUCTION
                                ;*****************************************************************************
                                ;INITIALIZE THE STACK POINTER
                                        ;SET UP NEW PC IN VECTOR
                                        ;SET UP NEW PSW IN VECTOR
                                ;CLEAR PRESENT (OLD) STATUS
                                ;DO TRAP
                                ;INSTRUCTION FAILED TO TRAP
                                ;IS NEW STATUS CORRECT
                                ;NEW STATUS WRONG
                                ;DID STACK DECREMENT CORRECTLY
                                ;STACK DID NOT DECREMENT CORRECTLY
                                ;WAS PROPER PC SAVED
                                ;PROPER PC WAS NOT SAVED
                                ;WAS OLD PSW SAVED
                                ;WRONG PSW SAVED
                                ;RESET STACK POINTER
                                        ;SET UP NEW PC IN VECTOR
                                        ;SET UP NEW PSW IN VECTOR
                                ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
                                ;DO TRAP WITH LOWER BYTE ALL ONES
                                ;INSTRUCTION FAILED TO TRAP
                                ;IS NEW PSW CORRECT
                                ;NEW PSW WRONG
                                ;WAS OLD STATUS STORED
                                ;OLD STATUS WRONG
                                ;WE MUST GET OVER ERROR CALL AT END OF THIS TEST
                                ;*****************************************************************************
                                ;THIS ERROR IS NEEDED BECAUSE BRANCHES IN TRAP TESTS BEYOND HERE CAN NOT
                                ;REACH ERROR1.
                                ;*****************************************************************************
                                ;*****************************************************************************
                                ;TEST 6 TEST TRAP OF TRACE-TRAP INSTRUCTION
                                ;*****************************************************************************
                                ;INITIALIZE THE STACK POINTER
                                        ;SET UP NEW PC IN VECTOR
                                        ;SET UP NEW PSW IN VECTOR
```

```
                                        CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
                                        TRT                     ;DO TRAP
                                8$:
                                        BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
                                1$:     CMP     R0,STATUS       ;IS NEW STATUS CORRECT
                                        BNE     ERROR2          ;NEW STATUS WRONG
                                2$:     CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
                                        BNE     ERROR2          ;STACK DID NOT DECREMENT CORRECTLY
                                3$:     CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
                                        BNE     ERROR2          ;PROPER PC WAS NOT SAVED
                                4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
                                        BNE     ERROR2          ;WRONG PSW SAVED
                                5$:     ADD     #4,SP           ;RESET STACK POINTER
                                        MOV     #6$,RTRAP4              ;SET UP NEW PC IN VECTOR
                                        CLR     RTRAP4+2                ;SET UP NEW PSW IN VECTOR
                                        MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
                                        TRT                     ;DO TRAP
                                        BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
                                6$:     TST     STATUS          ;IS NEW PSW CORRECT
                                        BNE     ERROR2          ;NEW PSW WRONG
                                7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
                                        BNE     ERROR2          ;OLD STATUS WRONG
                            ;PDP-11 ILLEGAL AND ADDRESS INSTRUCTION TEST
                            ;ALL INSTRUCTIONS THAT ARE RESERVED
                            ;SHOULD TRAP TO LOCATION 4, AND THE
                            ;PC THAT POINTS TO THE TRAPPING INSTRUCTION
                            ;SHOULD BE PLACED ON THE STACK

                            ;*****************************************************************************************
                            ;TEST 7 TEST TRAP OF ILLEGAL INSTRUCTION
                            ;*****************************************************************************************
                            TS7:
                                        MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
                                        MOV     #1$,RTRAP5              ;SET UP NEW PC IN VECTOR
                                        MOV     R0,RTRAP5+2             ;SET UP NEW PSW IN VECTOR
                                        CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
                                        JMP     %0                      ;DO TRAP
                                8$:
                                        BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
                                1$:     CMP     R0,STATUS       ;IS NEW STATUS CORRECT
                                        BNE     ERROR2          ;NEW STATUS WRONG
                                2$:     CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
                                        BNE     ERROR2          ;STACK DID NOT DECREMENT CORRECTLY
                                3$:     CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
                                        BNE     ERROR2          ;PROPER PC WAS NOT SAVED
                                4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
                                        BNE     ERROR2          ;WRONG PSW SAVED
                                5$:     ADD     #4,SP           ;RESET STACK POINTER
                                        MOV     #6$,RTRAP5              ;SET UP NEW PC IN VECTOR
                                        CLR     RTRAP5+2                ;SET UP NEW PSW IN VECTOR
                                        MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
                                        JMP     %0                      ;DO TRAP
                                        BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
                                6$:     TST     STATUS          ;IS NEW PSW CORRECT
                                        BNE     ERROR2          ;NEW PSW WRONG
                                7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
```

```
                        BNE     ERROR2          ;OLD STATUS WRONG
                ;*******************************************************************************
                ;TEST 10         TEST TRAP OF ALL ILLEGAL INSTRUCTION
                ;*******************************************************************************
                TS10:
002614 012706 001000            MOV     #STBOT,SP       ;INITIALIZE THE STACK POINTER
002620 012767 002642 175156     MOV     #1$,RTRAP5              ;SET UP NEW PC IN VECTOR
002626 010067 175154            MOV     R0,RTRAP5+2             ;SET UP NEW PSW IN VECTOR
002632 005067 175140            CLR     STATUS          ;CLEAR PRESENT (OLD) STATUS
002636 004000                   JSR     %0,%0                   ;DO TRAP
002640                  8$:
002640 000642                   BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
002642 020067 175130    1$:     CMP     R0,STATUS       ;IS NEW STATUS CORRECT
002646 001237                   BNE     ERROR2          ;NEW STATUS WRONG
002650 020627 000774    2$:     CMP     SP,#STBOT-4     ;DID STACK DECREMENT CORRECTLY
002654 001234                   BNE     ERROR2          ;STACK DID NOT DECREMENT CORRECTLY
002656 022767 002640 176110 3$: CMP     #8$,STBOT-4     ;WAS PROPER PC SAVED
002664 001230                   BNE     ERROR2          ;PROPER PC WAS NOT SAVED
002666 005767 176104    4$:     TST     STBOT-2         ;WAS OLD PSW SAVED
002672 001225                   BNE     ERROR2          ;WRONG PSW SAVED
002674 062706 000004    5$:     ADD     #4,SP           ;RESET STACK POINTER
002700 012767 002722 175076     MOV     #6$,RTRAP5              ;SET UP NEW PC IN VECTOR
002706 005067 175074            CLR     RTRAP5+2                ;SET UP NEW PSW IN VECTOR
002712 010037 177776            MOV     R0,@#STATUS     ;SET UP OLD STATUS FOR COMPARISON AFTER TRAP
002716 004000                   JSR     %0,%0                   ;DO TRAP
002720 000612                   BR      ERROR2          ;INSTRUCTION FAILED TO TRAP
002722 005767 175050    6$:     TST     STATUS          ;IS NEW PSW CORRECT
002726 001207                   BNE     ERROR2          ;NEW PSW WRONG
002730 020067 176042    7$:     CMP     R0,STBOT-2      ;WAS OLD STATUS STORED
002734 001204                   BNE     ERROR2          ;OLD STATUS WRONG

                ;*******************************************************************************
                .SBTTL  DATA PATH TESTS
                ;
                ;       THE DATA PATH TESTS ARE USED TO VERIFY THAT VARIOUS
                ;DATA PATTERNS CAN BE SUCCESSFULLY MOVED THROUGH THE DATA PATHS
                ;MOVE AND COMPARE MODE 2,3 INSTRUCTIONS ARE USED TO PASS AND
                ;TEST VARIOUS DATA PATTERNS IN THE DATA PATHS.
                ;       THE TEST EXERCISES THE INTERNAL DATA PATHS,  AND THE UNIBUS
                ;DATA TRANSCIEVERS.
                ;       IF THESE TESTS FAIL, EXAMINE THE TARGET LOCATION (LOC. 0)
                ;TO SEE WHICH BITS OF THE DATA PATH ARE FAILING.
002736 012737 002062 000030     MOV     #ERROR1,@#30    ;SET UP VECTOR FOR ERROR CALLS
002744 012737 000340 000032     MOV     #340,@#32       ;SET UP NEW PSW
002752 012737 021216 000004     MOV     #T04,@#4        ;SET UP FOR UNEXPECTED TRAP TO 4
002760 012737 021220 000010     MOV     #T010,@#10      ;SET UP FOR UNEXPECTED TRAP TO 10
002766 012737 021222 000014     MOV     #T014,@#14      ;SET UP FOR UNEXPECTED TRAP TO 14
002774 012737 021230 000034     MOV     #T034,@#34      ;SET UP FOR UNEXPECTED TRAP TO 34


                ;*******************************************************************************
                ;TEST 11         TEST OF ZEROES IN THE DATA PATH
                ;*******************************************************************************
                TS11:
003002 012737 000000 000000     MOV     #0,@#0          ;MOVE ZEROES THRU ADDRESS LINES, DATA
                                                        ;LINES AND INTERNAL PATHS
```

```
1002  003010  005737  000000            TST     @#0             ;SUCCESSFUL?
1003  003014  001401                    BEQ     TS12
1004  003016  104000                    EMT                     ;DATA INCORRECT
1005
1006                                    ;*********************************************************************************
1007                                    ;TEST 12        TEST OF PATTERN 125252 IN DATA PATH
1008                                    ;*********************************************************************************
1009  003020                            TS12:
1010  003020  012737  125252  000000    MOV     #125252,@#0     ;MOVE ALTERNATING ONES AND ZEROES
1011                                                            ;THRU DATA PATHS
1012  003026  022737  125252  000000    CMP     #125252,@#0     ;SUCCESSFUL
1013  003034  001401                    BEQ     TS13
1014  003036  104000                    EMT                     ;DATA INCORRECT
1015
1016                                    ;*********************************************************************************
1017                                    ;TEST 13        TEST OF PATTERN 052525 IN DATA PATH
1018                                    ;*********************************************************************************
1019  003040                            TS13:
1020  003040  012737  052525  000000    MOV     #052525,@#0     ;MOVE ALTERNATING ZEROES AND ONES
1021                                                            ;THRU DATA PATH
1022  003046  022737  052525  000000    CMP     #052525,@#0     ;SUCCESSFUL?
1023  003054  001401                    BEQ     TS14
1024  003056  104000                    EMT                     ;DATA INCORRECT
1025
1026                                    ;*********************************************************************************
1027                                    ;TEST 14        TEST OF ALL ONES IN DATA PATH
1028                                    ;*********************************************************************************
1029  003060                            TS14:
1030  003060  012737  177777  000000    MOV     #177777,@#0     ;MOVE ONES THRU DATA PATH
1031  003066  022737  177777  000000    CMP     #177777,@#0     ;SUCCESSFUL
1032  003074  001401                    BEQ     TS15
1033  003076  104000                    EMT                     ;DATA INCORRECT
1034
1035                                    ;*********************************************************************************
1036                                    .SBTTL  B-REGISTER TEST
1037                                    ;
1038                                    ;       THE B-REGISTER (LOCATION 0) SHIFTING LOGIC TESTS ARE USED
1039                                    ;TO TEST THAT THE B-REGISTER CAN HOLD VARIOUS DATA PATTERNS AND THAT
1040                                    ;THE ASSOCIATED LOGIC SUPPORTS THE SHIFTING FUNCTIONS WITHIN THE
1041                                    ;B-REGISTER AND C-BIT.
1042                                    ;A ONE IS SHIFTED THROUGH EVERY BIT IN THE B-REGISTER AND C-BIT IN
1043                                    ;BOTH DIRECTIONS.
1044                                    ;       THE B-REGISTER ITSELF IS TESTED IN ITS ABILITY AS A BUFFER AND AS
1045                                    ;A SHIFT REGISTER.  DATA IS ALSO PASSED THROUGH THE DATA PATH AND ALU.
1046                                    ;       IF THESE TESTS FAIL, EXAMINE THE TARGET LOCATION (LOC. 0) TO SEE
1047                                    ;WHICH BITS OF THE B-REGISTER MAY BE FAILING.
1048                                    ;
1049                                    ;*********************************************************************************
1050                                    ;TEST 15        SHIFT BIT 0 TO BIT 1
1051                                    ;*********************************************************************************
1052  003100                            TS15:
1053  003100  000241                    CLC                     ;CLEAR CARRY BIT
1054  003102  012737  000001  000000    MOV     #1,@#0          ;LOAD A 1
1055  003110  006137  000000            ROL     @#0             ;SHIFT LEFT
1056  003114  022737  000002  000000    CMP     #2,@#0          ;SUCCESSFUL
1057  003122  001401                    BEQ     TS16
```

```
1058  003124  104000                          EMT                     ;BIT 1 NOT SET
1059
1060                                          ;*************************************************************************************
1061                                          ;TEST 16        SHIFT CARRY INTO BIT 0
1062                                          ;*************************************************************************************
1063  003126                                  TS16:
1064  003126  012737  000000  000000                MOV     #0,@#0          ;CLEAR LOCATION
1065  003134  000261                          SEC                     ;SET CARRY
1066  003136  006137  000000                 ROL     @#0             ;ROTATE CARRY BIT TO BIT 0
1067  003142  103001                          BCC     CARRY1
1068  003144  104000                          EMT                     ;CARRY CLEAR
1069  003146  022737  000001  000000  CARRY1: CMP     #1,@#0          ;BIT 0 SET
1070  003154  001401                          BEQ     TS17
1071  003156  104000                          EMT                     ;BIT 0 NOT SET
1072
1073                                          ;*************************************************************************************
1074                                          ;TEST 17        LEFT SHIFT FROM BIT 0 TO C-BIT
1075                                          ;*************************************************************************************
1076  003160                                  TS17:
1077  003160  012737  000001  000000                MOV     #1,@#0          ;SET BIT 0
1078  003166  012700  177757                 MOV     #-21,R0         ;SET BIT COUNTER
1079  003172  000241                          CLC                     ;CLEAR C-BIT
1080  003174  005200                  SHL:    INC     R0              ;INCREMENT BIT COUNTER
1081  003176  001404                          BEQ     SHLE            ;BR TO ERROR HALT IF BIT IS LOST
1082  003200  006137  000000                 ROL     @#0             ;SHIFT LEFT ONE POSITION
1083  003204  103373                          BCC     SHL             ;BRANCH IF C-BIT NOT SET
1084  003206  001401                          BEQ     TS20
1085  003210                                  SHLE:
1086  003210  104000                          EMT                     ;LEFT SHIFTING LOGIC FAILED
1087
1088                                          ;*************************************************************************************
1089                                          ;TEST 20        SHIFT BIT 15 TO BIT 14
1090                                          ;*************************************************************************************
1091  003212                                  TS20:
1092  003212  012737  100000  000000                MOV     #100000,@#0     ;SET BIT 15
1093  003220  000241                          CLC                     ;CLEAR CARRY
1094  003222  006037  000000                 ROR     @#0             ;SHIFT BIT 15 TO BIT 14
1095  003226  022737  040000  000000                CMP     #40000,@#0      ;SUCCESSFUL
1096  003234  001401                          BEQ     TS21
1097  003236  104000                          EMT                     ;BIT 14 NOT SET
1098
1099                                          ;*************************************************************************************
1100                                          ;TEST 21        RIGHT SHIFT FROM BIT 15 TO C-BIT
1101                                          ;*************************************************************************************
1102  003240                                  TS21:
1103  003240  012737  100000  000000                MOV     #100000,@#0     ;SET BIT 15
1104  003246  012700  177757                 MOV     #-21,R0         ;SET BIT COUNTER
1105  003252  000241                          CLC                     ;CLEAR C-BIT
1106  003254  005200                  SHR:    INC     R0              ;INCREMENT BIT COUNTER
1107  003256  001404                          BEQ     SHRE            ;BR TO ERROR HALT IF BIT IS LOST
1108  003260  006037  000000                 ROR     @#0             ;ROTATE RIGHT ONE POSITION
1109  003264  103373                          BCC     SHR             ;BRANCH IF C-BIT CLEAR
1110  003266  001401                          BEQ     TS22
1111  003270                                  SHRE:
1112  003270  104000                          EMT                     ;RIGHT SHIFT LOGIC FAILED
1113
```

```
                        ;*****************************************************************************************
                        .SBTTL  SCRATCH PAD TESTS
                        ;
                        ;       THE SCRATCH PAD TESTS ARE USED TO VERIFY THAT VARIOUS
                        ;DATA PATTERNS CAN BE SUCCESSFULLY HELD IN THE SCRATCH PAD
                        ;CIRCUITRY.  MOVE AND COMPARE INSTRUCTIONS ARE USED TO TEST THAT
                        ;R0 CAN HOLD VARIOUS DATA PATTERNS.  EACH DATA PATTERN IS
                        ;MOVED AND TESTED IN A SMALL LOOP CONVENIENT FOR SCOPING.  THE
                        ;SUCCESSFUL COMPLETION OF THESE TESTS SHOULD VERIFY THE CIRCUITRY EXTERNAL
                        ;TO THE SCRATCH PAD ITSELF.
                        ;       THE REMAINDER OF THE GENERAL REGISTERS ARE TESTED BY MOVING
                        ;A BIT INTO BIT 0 OF THE REGISTER AND SHIFTING IT LEFT ONE
                        ;BIT AT A TIME INTO THE CARRY BIT.  THE RESULT IS THEN CHECKED TO INSURE THAT
                        ;NO BITS WERE PICKED. THE PROCEDURE IS THEN REPEATED UNDER OPPOSITE
                        ;CONDITIONS. THE GENERAL REGISTER AND THE CARRY BIT ARE SET TO
                        ;ALL ONES, AND A ZERO IS SHIFTED LEFT FROM BIT 0 INTO THE CARRY BIT.
                        ;THE RESULT IS THEN CHECKED TO INSURE THAT NO ZEROES WERE PICKED.
                        ;       AT THIS POINT ALL OF THE GENERAL REGISTERS HAVE BEEN EXERCISED
                        ;AS WELL AS REGISTER 11.
                        ;
                        ;*****************************************************************************************
                        ;TEST 22        TEST IF R0 CAN HOLD ALL ZEROES
                        ;*****************************************************************************************
003272                  TS22:

003272  012700  000000          MOV     #0,R0           ;MOVE ZEROES TO R0
003276  005700                  TST     R0              ;SUCCESSFUL?
003300  001401                  BEQ     TS23
003302  104000                  EMT                     ;R0 NOT 0

                        ;*****************************************************************************************
                        ;TEST 23        TEST IF R0 CAN HOLD ONES AND ZEROES
                        ;*****************************************************************************************
003304                  TS23:
003304  012700  125252          MOV     #125252,R0      ;MOVE ALTERNATING ONES AND ZEROES TO R0
003310  020027  125252          CMP     R0,#125252      ;SUCCESSFUL?
003314  001401                  BEQ     TS24
003316  104000                  EMT                     ;R0 NOT 125252

                        ;*****************************************************************************************
                        ;TEST 24        TEST IF R0 CAN HOLD ZEROES AND ONES
                        ;*****************************************************************************************
003320                  TS24:
003320  012700  052525          MOV     #052525,R0      ;MOVE ALTERNATING ZEROES AND ONES TO R0
003324  020027  052525          CMP     R0,#052525      ;SUCCESSFUL?
003330  001401                  BEQ     TS25
003332  104000                  EMT                     ;R0 NOT 52525

                        ;*****************************************************************************************
                        ;TEST 25        TEST IF R0 CAN HOLD ALL ONES
                        ;*****************************************************************************************
003334                  TS25:
003334  012700  177777          MOV     #177777,R0      ;MOVE ALL ONES TO R0
003340  020027  177777          CMP     R0,#177777      ;SUCCESSFUL?
003344  001401                  BEQ     TS26
003346  104000                  EMT                     ;R0 NOT 177777
```

```
1170
1171                    ;*****************************************************************************
1172                    ;TEST 26        TEST IF R1 CAN HOLD A ONE IN ALL BITS
1173                    ;*****************************************************************************
1174  003350            TS26:
1175  003350  012701  000001        MOV     #1,R1           ;SET BIT 0
1176  003354  012700  177757        MOV     #-21,R0         ;SET BIT COUNTER
1177  003360  000241                 CLC                     ;CLEAR C-BIT
1178  003362  005200        REG1:    INC     R0              ;INCREMENT BIT COUNTER
1179  003364  001403                 BEQ     REG1E           ;BR TO ERROR HALT IF BIT IS LOST
1180  003366  006101                 ROL     R1              ;ROTATE 1 POSITION
1181  003370  103374                 BCC     REG1            ;ALL DONE
1182  003372  001401                 BEQ     TS27
1183  003374                 REG1E:
1184  003374  104000                 EMT                     ;FAILURE WITH R1
1185
1186                    ;*****************************************************************************
1187                    ;TEST 27        TEST IF R1 CAN HOLD A ZERO IN ALL BITS
1188                    ;*****************************************************************************
1189  003376            TS27:
1190  003376  012701  177776        MOV     #-2,R1          ;SET ALL ONES IN R1 EXCEPT FOR BIT 0
1191  003402  012700  177757        MOV     #-21,R0         ;SET BIT COUNTER
1192  003406  000261                 SEC                     ;SET C-BIT
1193  003410  005200        REG1A:   INC     R0              ;INCREMENT COUNTER
1194  003412  001405                 BEQ     R1ERR           ;BR TO ERROR HALT IF COUNTER=0
1195  003414  006101                 ROL     R1              ;ROTATE 1 POSITION
1196  003416  103774                 BCS     REG1A           ;CONTINUE UNTIL C-BIT IS CLEAR
1197  003420  022701  177777        CMP     #-1,R1          ;CHECK DATA IN R1
1198  003424  001401                 BEQ     TS30
1199  003426                 R1ERR:
1200  003426  104000                 EMT                     ;FAILURE WITH R1
1201                    ;*****************************************************************************
1202                    ;TEST 30        TEST IF R2 CAN HOLD A ONE IN ALL BITS
1203                    ;*****************************************************************************
1204  003430            TS30:
1205  003430  012702  000001        MOV     #1,R2           ;SET BIT 0
1206  003434  012700  177757        MOV     #-21,R0         ;SET BIT COUNTER
1207  003440  000241                 CLC                     ;CLEAR C-BIT
1208  003442  005200        REG2:    INC     R0              ;INCREMENT BIT COUNTER
1209  003444  001403                 BEQ     REG2E           ;BR TO ERROR HALT IF BIT IS LOST
1210  003446  006102                 ROL     R2              ;ROTATE 1 POSITION
1211  003450  103374                 BCC     REG2            ;ALL DONE
1212  003452  001401                 BEQ     TS31
1213  003454                 REG2E:
1214  003454  104000                 EMT                     ;FAILURE WITH R2
1215
1216                    ;*****************************************************************************
1217                    ;TEST 31        TEST IF R2 CAN HOLD A ZERO IN ALL BITS
1218                    ;*****************************************************************************
1219  003456            TS31:
1220  003456  012702  177776        MOV     #-2,R2          ;SET ALL ONES IN R2 EXCEPT FOR BIT 0
1221  003462  012700  177757        MOV     #-21,R0         ;SET BIT COUNTER
1222  003466  000261                 SEC                     ;SET C-BIT
1223  003470  005200        REG2B:   INC     R0              ;INCREMENT BIT COUNTER
1224  003472  001405                 BEQ     R2ERR           ;BR TO ERROR HALT IF COUNTER=0
1225  003474  006102                 ROL     R2              ;ROTATE 1 POSITION
```

```
1226  003476  103774                  BCS     REG2B           ;CONTINUE UNTIL C-BIT IS CLEAR
1227  003500  022702  177777          CMP     #-1,R2          ;CHECK DATA IN R2
1228  003504  001401                  BEQ     TS32
1229  003506                  R2ERR:
1230  003506  104000                  EMT                     ;FAILURE WITH R2
1231
1232          ;*****************************************************************************************
1233          ;TEST 32        TEST IF R3 CAN HOLD A ONE IN ALL BITS
1234          ;*****************************************************************************************
1235  003510                  TS32:
1236  003510  012703  000001          MOV     #1,R3           ;SET BIT 0
1237  003514  012700  177757          MOV     #-21,R0         ;SET BIT COUNTER
1238  003520  000241                  CLC                     ;CLEAR C-BIT
1239  003522  005200          REG3:   INC     R0              ;INCREMENT BIT COUNTER
1240  003524  001403                  BEQ     REG3E           ;BR TO ERROR HALT IF BIT IS LOST
1241  003526  006103                  ROL     R3              ;ROTATE 1 POSITION
1242  003530  103374                  BCC     REG3            ;ALL DONE
1243  003532  001401                  BEQ     TS33
1244  003534                  REG3E:
1245  003534  104000                  EMT                     ;FAILURE WITH R3
1246
1247          ;*****************************************************************************************
1248          ;TEST 33        TEST IF R3 CAN HOLD A ZERO IN ALL BITS
1249          ;*****************************************************************************************
1250  003536                  TS33:
1251  003536  012703  177776          MOV     #-2,R3          ;SET ALL ONES IN R3 EXCEPT FOR BIT 0
1252  003542  012700  177757          MOV     #-21,R0         ;SET BIT COUNTER
1253  003546  000261                  SEC                     ;SET C-BIT
1254  003550  005200          REG3A:  INC     R0              ;INCREMENT BIT COUNTER
1255  003552  001405                  BEQ     R3ERR           ;BR TO ERROR HALT IF COUNTER=0
1256  003554  006103                  ROL     R3              ;ROTATE 1 POSITION
1257  003556  103774                  BCS     REG3A           ;CONTINUE UNTIL C-BIT IS CLEAR
1258  003560  022703  177777          CMP     #-1,R3          ;CHECK DATA
1259  003564  001401                  BEQ     TS34
1260  003566                  R3ERR:
1261  003566  104000                  EMT                     ;FAILURE WITH R3
1262
1263          ;*****************************************************************************************
1264          ;TEST 34        TEST IF R4 CAN HOLD A ONE IN ALL BITS
1265          ;*****************************************************************************************
1266  003570                  TS34:
1267  003570  012704  000001          MOV     #1,R4           ;SET BIT 0
1268  003574  012700  177757          MOV     #-21,R0         ;SET BIT COUNTER
1269  003600  000241                  CLC                     ;CLEAR C-BIT
1270  003602  005200          REG4:   INC     R0              ;INCREMENT BIT COUNTER
1271  003604  001403                  BEQ     REG4E           ;BR TO ERROR HALT IF BIT IS LOST
1272  003606  006104                  ROL     R4              ;ROTATE 1 POSITION
1273  003610  103374                  BCC     REG4            ;ALL DONE
1274  003612  001401                  BEQ     TS35
1275  003614                  REG4E:
1276  003614  104000                  EMT                     ;FAILURE WITH R4
1277
1278          ;*****************************************************************************************
1279          ;TEST 35        TEST IF R4 CAN HOLD A ZERO IN ALL BITS
1280          ;*****************************************************************************************
1281  003616                  TS35:
```

```
1282  003616  012704  177776          MOV     #-2,R4          ;SET ALL ONES IN R4 EXCEPT FOR BIT 0
1283  003622  012700  177757          MOV     #-21,R0         ;SET BIT COUNTER
1284  003626  000261                  SEC                     ;SET C-BIT
1285  003630  005200          REG4A:  INC     R0              ;INCREMENT BIT COUNTER
1286  003632  001405                  BEQ     R4ERR           ;BR TO ERROR HALT IF COUNTER=0
1287  003634  006104                  ROL     R4              ;ROTATE 1 POSITION
1288  003636  103774                  BCS     REG4A           ;CONTINUE UNTIL C-BIT IS CLEAR
1289  003640  022704  177777          CMP     #-1,R4          ;CHECK DATA
1290  003644  001401                  BEQ     T36
1291  003646                  R4ERR:
1292  003646  104000                  EMT                     ;FAILURE WITH R4
1293
1294
1295                            ;*****************************************************************************
1296                            ;TEST 36        TEST IF R5 CAN HOLD A ONE IN ALL BITS
1297                            ;*****************************************************************************
1298  003650                  T36:
1299  003650  012705  000001          MOV     #1,R5           ;SET BIT 0
1300  003654  012700  177757          MOV     #-21,R0         ;SET BIT COUNTER
1301  003660  000241                  CLC                     ;CLEAR C-BIT
1302  003662  005200          REG5:   INC     R0              ;INCREMENT BIT COUNTER
1303  003664  001403                  BEQ     REG5E           ;BR TO ERROR HALT IF BIT IS LOST
1304  003666  006105                  ROL     R5              ;ROTATE 1 POSITION
1305  003670  103374                  BCC     REG5            ;ALL DONE
1306  003672  001401                  BEQ     T37
1307  003674                  REG5E:
1308  003674  104000                  EMT                     ;FAILURE WITH R5
1309
1310                            ;*****************************************************************************
1311                            ;TEST 37        TEST IF R5 CAN HOLD A ZERO IN ALL BITS
1312                            ;*****************************************************************************
1313  003676                  T37:
1314  003676  012705  177776          MOV     #-2,R5          ;SET ALL ONES IN R5 EXCEPT FOR BIT 0
1315  003702  012700  177757          MOV     #-21,R0         ;SET BIT COUNTER
1316  003706  000261                  SEC                     ;SET C-BIT
1317  003710  005200          REG5A:  INC     R0              ;INCREMENT BIT COUNTER
1318  003712  001405                  BEQ     R5ERR           ;BR TO ERROR HALT IF COUNTER=0
1319  003714  006105                  ROL     R5              ;ROTATE 1 POSITION
1320  003716  103774                  BCS     REG5A           ;CONTINUE UNTIL C-BIT IS C;EAR
1321  003720  022705  177777          CMP     #-1,R5          ;CHECK DATA
1322  003724  001401                  BEQ     T40
1323  003726                  R5ERR:
1324  003726  104000                  EMT                     ;FAILURE WITH R5
1325
1326                            ;*****************************************************************************
1327                            ;TEST 40        TEST IF R6 CAN HOLD A ONE IN ALL BITS
1328                            ;*****************************************************************************
1329  003730                  T40:
1330  003730  012706  000001          MOV     #1,R6           ;SET BIT 0
1331  003734  012700  177757          MOV     #-21,R0         ;SET BIT COUNTER
1332  003740  000241                  CLC                     ;CLEAR C-BIT
1333  003742  005200          REG6:   INC     R0              ;INCREMENT BIT COUNTER
1334  003744  001403                  BEQ     REG6E           ;BR TO ERROR HALT IF BIT IS LOST
1335  003746  006106                  ROL     R6              ;ROTATE 1 POSITION
1336  003750  103374                  BCC     REG6            ;ALL DONE
1337  003752  001401                  BEQ     T41
```

```
1338  003754
1339  003754  104000                    REG6E:
1340                                            EMT                     ;FAILURE WITH R6
1341
1342                                    ;*******************************************************************************
1343                                    ;TEST 41        TEST IF R6 CAN HOLD A ZERO IN ALL BITS
1344  003756                            ;*******************************************************************************
1345  003756  012706  177776            TS41:
1346  003762  012700  177757                   MOV     #-2,R6          ;SET ALL ONES IN R6 EXCEPT FOR BIT 0
1347  003766  000261                           MOV     #-21,R0         ;SET BIT COUNTER
1348  003770  005200                           SEC                     ;SET C-BIT
1349  003772  001405            REG6A:  INC     R0              ;INCREMENT BIT COUNT
1350  003774  006106                           BEQ     R6ERR           ;BR TO ERROR HALT IF COUNTER=0
1351  003776  103774                           ROL     R6              ;ROTATE 1 POSITION
1352  004000  022706  177777           BCS     REG6A           ;CONTINUE UNTIL C-BIT IS CLEAR
1353  004004  001401                   CMP     #-1,R6          ;CHECK DATA
1354  004006                           BEQ     TS42
1355  004006  104000            R6ERR:
1356                                           EMT                     ;FAILURE WITH R6
1357
1358                                    ;*******************************************************************************
1359                                    .SBTTL  PSW TESTS
1360                                    ;
1361                                    ;       THE PSW TESTS ARE USED TO VERIFY THAT VARIOUS DATA
1362                                    ;PATTERNS CAN BE SUCCESSFULLY HELD IN THE PSW AND THAT THE
1363                                    ;PSW ADDRESSING LOGIC IS FUNCTIONING.  MOVE AND COMPARE INSTRUCTIONS
1364                                    ;ARE USED TO TEST THAT THE PSW CAN HOLD VARIOUS DATA PATTERNS.
1365                                    ;EACH DATA PATTERN IS MOVED AND TESTED IN A SMALL LOOP CONVENIENT FOR
1366                                    ;SCOPING.
1367                                    ;       THE PSW REGISTER IS TESTED, THE CC INPUTS ARE TESTED
1368                                    ;LATER IN THE MICROCODE TESTS.  SETTING OF THE T-BIT BY THE
1369                                    ;TEST PATTERNS IS PURPOSELY AVOIDED. TESTING OF THE
1370                                    ;T-BIT TRAP CIRCUITRY IS LEFT FOR THE TRAP TEST.
1371                                    ;
1372                                    ;*******************************************************************************
1373                                    ;TEST 42        TEST IF PSW WILL HOLD ZEROES
1374  004010                            ;*******************************************************************************
1375  004010  012706  001000            TS42:
1376  004014  012737  000000  177776           MOV     #STBOT,R6
1377  004022  005737  177776                   MOV     #0,@#PS         ;SET PSW TO ZERO
1378  004026  001401                           TST     @#PS            ;SUCCESSFUL
1379  004030  104000                           BEQ     TS43
1380                                           EMT                     ;PSW NOT 0
1381
1382                                    ;*******************************************************************************
1383                                    ;TEST 43        TEST IF PSW WILL HOLD ONES AND ZEROES
1384  004032                            ;*******************************************************************************
1385  004032  012737  000252  177776    TS43:
1386  004040  023727  177776  000252           MOV     #252,@#PS       ;MOVE ALT. ONES AND ZEROES TO PSW
1387  004046  001401                           CMP     @#PS,#252       ;SUCCESSFUL?
1388  004050  104000                           BEQ     TS44
1389                                           EMT                     ;PSW NOT 252
1390
1391                                    ;*******************************************************************************
1392                                    ;TEST 44        TEST IF PSW (EXCEPT T-BIT) WILL HOLD ZEROES AND ONES
1393  004052                            ;*******************************************************************************
                                        TS44:
```

```
1394  004052  012737  000105  177776          MOV     #105,@#PS       ;MOVE ALT. ONES AND ZEROES TO PSW
1395  004060  023727  177776  000105          CMP     @#PS,#105       ;SUCCESSFUL?
1396  004066  001401                          BEQ     TS45
1397  004070  104000                          EMT                     ;PSW NOT 105
1398
1399                                  ;*******************************************************************************
1400                                  ;TEST 45        TEST IF PSW (EXCEPT T-BIT) WILL HOLD ALL ONES
1401                                  ;*******************************************************************************
1402  004072                          TS45:
1403  004072  012737  000357  177776          MOV     #357,@#PS       ;MOVE ONES TO PSW
1404  004100  023727  177776  000357          CMP     @#PS,#357       ;SUCCESSFUL
1405  004106  001401                          BEQ     TS46
1406  004110  104000                          EMT                     ;PSW NOT 357
1407                                  ;*******************************************************************************
1408                                  .SBTTL  MICROCODE TESTS
1409                                  ;
1410                                  ;       THE TEST EXERCISES  BRANCHES IN THE MICROCODE BY
1411                                  ;TESTING AT LEAST ONE INSTRUCTION FROM EVERY CLASS OF INSTRUCTION IN
1412                                  ;ALL POSSIBLE MODES.  FOR EXAMPLE, TO TEST THE SINGLE OPERAND INSTRUCTIONS,
1413                                  ;AT LEAST ONE SINGLE OPERAND INSTRUCTION IS VERIFIED IN ALL UNIQUE
1414                                  ;ADDRESSING MODES.  BYTE MODES ARE ALSO TESTED.  AS EACH NEW
1415                                  ;MODE IS INTRODUCED THE SAME INSTRUCTION IS TRIED AND TESTED IN
1416                                  ;A SMALL LOOP CONVENIENT FOR SCOPING.  THE TEST IS SET UP USING
1417                                  ;ONLY INSTRUCTIONS AND ADDRESSING MODES WHICH HAVE BEEN PREVIOUSLY
1418                                  ;VERIFIED.
1419                                  ;       IF THESE TESTS FAIL, CHECK THE RESULTS FOR A CLUE TO THE
1420                                  ;FAULT.
1421                                  ;
1422                                  ;*******************************************************************************
1423
1424
1425
1426                                  ;*******************************************************************************
1427                                  ;
1428                                  ;       THE CLR INSTRUCTION IS USED TO INTRODUCE EACH ADDRESSING
1429                                  ;MODE WITH THE SINGLE OPERAND INSTRUCTION.  FOLLOWING THE SEQUENCE CHECK,
1430                                  ;THE CLR INSTRUCTION IS EXECUTED AND A BRANCH TEST IS EXECUTED WHICH
1431                                  ;CHECKS THAT THE Z-BIT WAS PROPERLY SET.  THIS TEST CAN CHECK IR DECODE
1432                                  ;AND MICROCODE FOR SOP INSTRUCTIONS WITH MODE 0.  FOLLOWING THIS TEST
1433                                  ;SEVERAL OTHER SOP INSTRUCTIONS ARE INTRODUCED WITH MODE 0.  THESE
1434                                  ;INSTRUCTIONS MAINPULATE DATA AND SERVE TO CHECK THE DATA RESULTS
1435                                  ;OF THE SOP INSTRUCTIONS IN THIS TEST.  THE DATA IN THIS TEST IS
1436                                  ;OPERATED ON BY EACH INSTRUCTION WITHOUT REINITIALIZING.
1437                                  ;
1438                                  ;*******************************************************************************
1439                                  ;TEST 46        TEST MODE 0 USING SOP INST.
1440                                  ;*******************************************************************************
1441  004112                          TS46:
1442  004112  005000                          CLR     R0              ;TRY THE CLEAR INST.
1443  004114  001401                          BEQ     SOP0A
1444  004116  104000                          EMT                     ;CLR DID NOT SET Z-BIT
1445  004120  005200                  SOP0A:  INC     R0              ;TRY THE INCREMENT INST.
1446  004122  005100                          COM     R0              ;TRY COMPLEMENT
1447  004124  005200                          INC     R0
1448  004126  100401                          BMI     SOP0B
1449  004130  104000                          EMT                     ;NEGATE DID NOT SET N-BIT
```

```
1450  004132  005100          SOP0B:  COM     R0              ;TRY COMPLEMENT INST.
1451  004134  001401                  BEQ     TS47
1452  004136  104000                  EMT                     ;CUMMULATIVE RESULT OF CLR,INC,NEG AND COM INSTS. FAILED
1453
1454                          ;*********************************************************************************************
1455                          ;
1456                          ;       THIS TEST INTRODUCES THE REMAINING SOP INSTRUCIONS AND TESTS
1457                          ;THEM IN MODE 0.  THE PURPOSE IS TO PROVIDE A BASELINE OF
1458                          ;INSTRUCTIONS FOR USE IN THE SUBSEQUENT TESTS.  SINCE THE MICROCODE FOR
1459                          ;THESE INSTRUCTIONS IS IDENTICAL TO THAT ALREADY TESTED, ANY TROUBLE
1460                          ;SHOOTING EFFORTS SHOULD BE AIMED AT THE ACTUAL IR DECODE AND ALU
1461                          ;FUNCTIONING.
1462                          ;
1463                          ;*********************************************************************************************
1464                          ;TEST 47        TEST REMAINDER OF SOP INSTS IN MODE 0
1465                          ;*********************************************************************************************
1466                          TS47:
1467  004140                          CLR     R0              ;INITIALIZE
1468  004140  005000                  DEC     R0              ;TRY DECREMENT INST.
1469  004142  005300                  BMI     SOP0C
1470  004144  100401                  EMT                     ;N-BIT NOT SET ON DEC
1471  004146  104000          SOP0C:  SEC                     ;INITIALIZE CARRY
1472  004150  000261                  ADC     R0              ;TRY ADD CARRY INST
1473  004152  005500                  BNE     SOP0D
1474  004154  001007                  SEC                     ;INITIALIZE CARRY
1475  004156  000261                  SBC     R0              ;TRY SUBTRACT-CARRY INST
1476  004160  005600                  BPL     SOP0D
1477  004162  100004                  COM     R0
1478  004164  005100                  INC     R0
1479  004166  005200                  DEC     R0
1480  004170  005300                  BEQ     TS50
1481  004172  001401          SOP0D:
1482  004174                          EMT                     ; CUMMULATIVE RESULT OF ADC,SBC,COM,INC AND DEC INSTS. F
1483  004174  104000
1484                          ;*********************************************************************************************
1485                          ;
1486                          ;       THIS TEST INTRODUCES THE BYTE CONTROL LOGIC OF THE PROCESSOR.
1487                          ;THE MODE 0 BYTE MICROCODE IS TESTED.  THE METHOD AND SEQUENCE
1488                          ;OF TESTING IS THE SAME AS THAT USED IN THE SOP MODE 0 TESTS.
1489                          ;
1490                          ;*********************************************************************************************
1491                          ;TEST 50        TEST MODE 0 EVEN BYTE USING SOP INST
1492                          ;*********************************************************************************************
1493                          TS50:
1494  004176                          CLRB    R0              ;TRY CLEARING EVEN BYTE OF REGISTER
1495  004176  105000                  BEQ     SOPB0A
1496  004200  001401                  EMT                     ;CLRB DID NOT SET Z-BIT
1497  004202  104000          SOPB0A: COMB    R0              ;TRY SETTING EVEN BYTE OF REGISTER
1498  004204  105100                  BPL     SOPB0B
1499  004206  100002                  INCB    R0              ;TRY INCREMENTING EVEN BYTE OF REGISTER>>
1500  004210  105200                  BEQ     TS51
1501  004212  001401          SOPB0B:
1502  004214                          EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE BYTE INST.
1503  004214  104000
1504                          ;*********************************************************************************************
1505
```

```
                        ;
                        ;       THIS TEST USES THE CLR INSTRUCTION TO INTRODUCE AND TEST
                        ;SINGLE OPERAND MODE 1 INSTRUCTIONS.  AGAIN, THE CLR INSTRUCTION
                        ;IS USED TO INTRODUCE THE MICROCODE AND TO TEST THAT THE PROPER
                        ;CONDITION CODES ARE SET.  OTHER SOP INSTRUCTIONS ARE USED TO MANIPULATE
                        ;COMMON DATA TO VERIFY THAT THE CORRECT DATA IS PRODUCED.
                        ;
                        ;*****************************************************************************
                        ;TEST 51        TEST MODE 1 USING SOP INST.
                        ;*****************************************************************************
004216                  TS51:
004216  005000                  CLR     R0              ;INITIALIZE R0
004220  005010                  CLR     (R0)            ;TRY CLEAR INST W/MODE 1
004222  001401                  BEQ     SOP1A
004224  104000                  EMT                     ;CLR DID NOT SET Z-BIT
004226  005310          SOP1A:  DEC     (R0)            ;TRY DECREMENT INST W/MODE 1
004230  100003                  BPL     SOP1B
004232  000261                  SEC                     ;INITIALIZE CARRY
004234  005510                  ADC     (R0)            ;TRY ADD-CARRY W/MODE 1
004236  001401                  BEQ     TS52
004240                  SOP1B:
004240  104000                  EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INST

                        ;*****************************************************************************
                        ;
                        ;       THIS TEST VERIFIES THE BYTE INSTRUCTION MICROCODE FOR MODE 1
                        ;SINGLE OPERAND INSTRUCTIONS.
                        ;       THIS IS THE FIRST PLACE THE SIGN EXTEND LOGIC IS EXERCISED
                        ;AND VERIFIED.
                        ;
                        ;*****************************************************************************
                        ;TEST 52        TEST MODE 1 EVEN BYTE USING SOP INST
                        ;*****************************************************************************
004242                  TS52:
004242  005000                  CLR     R0              ;INITIALIZE R0
004244  005010                  CLR     (R0)            ;INITIALIZE LOC. 0
004246  005110                  COM     (R0)
004250  105010                  CLRB    (R0)            ;TRY TO CLEAR BYTE 0
004252  001401                  BEQ     SOPB1A
004254  104000                  EMT                     ;CLRB DID NOT SET Z-BIT
004256  005210          SOPB1A: INC     (R0)            ;INCREMENT TO TEST WORD
004260  100005                  BPL     SOPB1B
004262  105110                  COMB    (R0)            ;COMPLEMENT:  ODD BYTE = 376
004264  105210                  INCB    (R0)            ;INC:  ODD BYTE = 377
004266  100002                  BPL     SOPB1B
004270  105210                  INCB    (R0)            ;INCREMENT ODD BYTE=0
004272  001401                  BEQ     TS53
004274                  SOPB1B:
004274  104000                  EMT                     ;CHECK CUMMULATIVE RESULT OF ABOVE INST


                        ;*****************************************************************************
                        ;
                        ;       THIS TEST VERIFIES THAT SINGLE OPERAND BYTE INSTRUCTIONS WILL
                        ;FUNCTION CORRECTLY FOR ODD BYTES.
                        ;       THIS IS THE FIRST TIME THAT ADDRESS LINE 0 HAS BEEN
```

```
1562
1563
1564
1565
1566
1567
1568
1569  004276
1570  004276  005000
1571  004300  005010
1572  004302  005110
1573  004304  005200
1574  004306  105010
1575  004310  001401
1576  004312  104000
1577  004314  005300
1578  004316  005210
1579  004320  005200
1580  004322  105110
1581  004324  105210
1582  004326  100002
1583  004330  105210
1584  004332  001401
1585  004334
1586  004334  104000
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601  004336
1602  004336  005000
1603  004340  105100
1604  004342  005200
1605  004344  005010
1606  004346  005110
1607  004350  005020
1608  004352  001401
1609  004354  104000
1610  004356  005300
1611  004360  005300
1612  004362  005120
1613  004364  100004
1614  004366  005300
1615  004370  005300
1616  004372  005220
1617  004374  001401
```

```
;EXERCISED.  CHECKS ARE MADE THAT THE PROPER BYTE IS MODIFIED AND
;THE CONDITION CODES ARE CHECKED.  IT IS ALSO VERIFIED THAT THE UNADDRESSED
;BYTE IS NOT ALTERED BY THE INSTRUCTION.
;
;*****************************************************************************
;TEST 53        TEST MODE 1 ODD BYTE USING SOP INST
;*****************************************************************************
TS53:
        CLR     R0              ;INITIALIZE R0
        CLR     (R0)            ;INITIALIZE LOC. 0
        COM     (R0)
        INC     R0              ;R0=ODD BYTE
        CLRB    (R0)            ;TRY TO CLEAR BYTE 1
        BEQ     SOPB1C
        EMT                     ;CLRB DID NOT SET Z-BIT
SOPB1C: DEC     R0              ;R0=WORD ADDR.
        INC     (R0)            ;INCREMENT TO TEST WORD
        INC     R0              ;R0=ODD BYTE
        COMB    (R0)            ;TRY TO COMPLEMENT BYTE 1
        INCB    (R0)
        BPL     SOPB1D
        INCB    (R0)            ;TRY TO INCREMENT BYTE 1
        BEQ     TS54
SOPB1D:
        EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INST.

;*****************************************************************************
;
;       THIS TEST VERIFIES MODE 2 SINGLE-OPERAND INSTRUCTIONS.  PREVIOUSLY
;TESTED INSTRUCTIONS ARE USED TO SET A POINTER IN R0 TO LOC. 400.
;LOC. 400 IS INITIALIZED TO -1 BEFORE A CLR MODE 2 IS EXECUTED.
;       THEN R0 IS DECREMENTED BY TWO TO AGAIN POINT TO 400 BEFORE EACH
;OF SEVERAL MODE 2 INSTRUCTIONS ARE USED TO VERIFY THE DATA RESULTS OF
;THE TEST.  THIS PROCEDURE ALSO VERIFIES THE PROPER INCREMENTING OF THE
;REGISTER.
;
;*****************************************************************************
;TEST 54        TEST MODE 2 USING SOP INST.
;*****************************************************************************
TS54:
        CLR     R0              ;SET R0=400
        COMB    R0
        INC     R0
        CLR     (R0)            ;CLEAR 400
        COM     (R0)            ;INITIALIZE: 400=-1
        CLR     (R0)+           ;TRY CLEARING WITH MODE 2
        BEQ     SOPZA
        EMT                     ;CLR INST DID NOT SET Z-BIT
SOPZA:  DEC     R0              ;RESET R0
        DEC     R0
        COM     (R0)+           ;TRY COMPLEMENTING WITH MODE 2
        BPL     SOP2B
        DEC     R0              ;RESET R0
        DEC     R0
        INC     (R0)+           ;TRY INCREMENTING WITH MODE 2
        BEQ     TS55
```

```
1618  004376
1619  004376  104000
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634  004400
1635  004400  005000
1636  004402  105100
1637  004404  005200
1638  004406  005010
1639  004410  005110
1640  004412  105020
1641  004414  001401
1642  004416  104000
1643  004420  005300
1644  004422  005210
1645  004424  105110
1646  004426  105220
1647  004430  100003
1648  004432  005300
1649  004434  105220
1650  004436  001401
1651  004440
1652  004440  104000
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662  004442
1663  004442  005000
1664  004444  105100
1665  004446  005200
1666  004450  005010
1667  004452  005110
1668  004454  005200
1669  004456  105020
1670  004460  001401
1671  004462  104000
1672  004464  005300
1673  004466  005300
```

```
SOP2B:
        EMT                     ;CHECK CUMMULATIVE RESULT OF ABOVE INST

;*******************************************************************************
;
;       THIS TEST VERIFIES MODE 2 SINGLE OPERAND INSTRUCTIONS WHICH
;ADDRESS EVEN BYTES.  R0 IS SET TO 400 AND USED TO INITIALIZE LOCATION
;400 TO -1.  CLRB INSTRUCTION IS THEN EXECUTED ON BYTE 400 WITH
;MODE 2.
;       R0 IS THEN DECREMENTED BEFORE EACH OF SEVERAL MODE 2 INSTRUCTIONS
;WHICH ARE USED TO VERIFY THE DATA RESULTS OF THE TEST.  THIS PROCEDURE ALSO
;VERIFIES THE PROPER INCREMENTING OF THE REGISTER.
;
;*******************************************************************************
;TEST 55        TEST MODE 2 EVEN BYTE USING SOP INST.
;*******************************************************************************
TS55:
        CLR     R0              ;SET R0=400
        COMB    R0
        INC     R0
        CLR     (R0)            ;CLEAR 400
        COM     (R0)            ;INITIALIZE: 400=-1
        CLRB    (R0)+           ;TRY TO CLEAT 400 W/MODE 2
        BEQ     SOPB2A
        EMT                     ;CLR DID NOT SET Z-BIT
SOPB2A: DEC     R0              ;RESULT R0=400
        INC     (R0)            ;INC 400 TO TEST WORD
        COMB    (R0)
        INCB    (R0)+           ;TRY TO INC EVEN BYTE
        BPL     SOPB2B
        DEC     R0              ;RESET R0=400
        INCB    (R0)+           ;TRY INCREMENT OF EVEN BYTE
        BEQ     TS56
SOPB2B:
        EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INST.

;*******************************************************************************
;
;       THIS TEST FOLLOWS THE SAME PROCEDURE DESCRIBED IN THE PREVIOUS
;TEST.  HERE, THE BYTE INSTRUCTION IS USED TO ADDRESS AN ODD BYTE.
;
;*******************************************************************************
;TEST 56        TEST MODE 2 ODD BYTE USING SOP INST.
;*******************************************************************************
TS56:
        CLR     R0              ;SET R0=400
        COMB    R0
        INC     R0
        CLR     (R0)            ;CLEAR LOC 400
        COM     (R0)            ;INITIALIZE: 400=-1
        INC     R0              ;R0=ODD BYTE
        CLRB    (R0)+           ;TRY TO CLEAR ODD BYTE
        BEQ     SOPB2C
        EMT                     ;CLRB DID NOT SET Z-BIT
SOPB2C: DEC     R0              ;R0=WORD ADDR.
        DEC     R0
```

```
1674  004470  005220                INC     (R0)+           ;INCREMENT WORD
1675  004472  005300                DEC     R0              ;POINT TO ODD BYTE
1676  004474  105110                COMB    (R0)            ;COMPLEMENT ODD BYTE
1677  004476  105220                INCB    (R0)+           ;TRY TO INCREMENT ODD BYTE
1678  004500  100003                BPL     SOPB2D
1679  004502  005300                DEC     R0              ;RESET R0 TO ODD BYTE
1680  004504  105220                INCB    (R0)+           ;TRY TO INCREMENT ODD BYTE
1681  004506  001401                BEQ     TS57
1682  004510                      SOPB2D:
1683  004510  104000                EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INST.
1684
1685                               ;*****************************************************************************
1686                               ;
1687                               ;       THESE TESTS CHECK THE NEGATE INSTRUCTION IN ALL MODES.  PREVIOUSLY
1688                               ;TESTED SINGLE-OPERAND INSTRUCTIONS ARE USED TO TEST THE NEGATE INSTRUCTION.
1689                               ;
1690                               ;*****************************************************************************
1691                               ;TEST 57        TEST MODE 0 USING NEGATE INSTRUCTION
1692                               ;*****************************************************************************
1693  004512                      TS57:
1694  004512  005000                CLR     R0              ;SET R0=0
1695  004514  005200                INC     R0              ;    R0=1
1696  004516  005400                NEG     R0              ;TRY NEGATE MODE 0:  R0=-1
1697  004520  100003                BPL     NEG00           ;CC=1001?
1698  004522  001402                BEQ     NEG00
1699  004524  102401                BVS     NEG00
1700  004526  103401                BCS     NEG01
1701  004530                      NEG00:
1702  004530  104000                EMT                     ;NEGATE DID NOT SET CC'S CORRECTLY
1703
1704  004532  005200              NEG01:  INC     R0              ;TEST DATA RESULT
1705  004534  001401                BEQ     NEG02
1706  004536  104000                EMT                     ;DATA RESULT OF NEGATE INCORRECT
1707
1708  004540  105100              NEG02:  COMB    R0              ;R0=377
1709  004542  105400                NEGB    R0              ;R0=1
1710  004544  100403                BMI     NEG03           ;CC=0001?
1711  004546  001402                BEQ     NEG03
1712  004550  102401                BVS     NEG03
1713  004552  103401                BCS     NEG04
1714  004554                      NEG03:
1715  004554  104000                EMT                     ;NEGB DID NOT SET CC'S CORRECTLY
1716  004556  005300              NEG04:  DEC     R0              ;TEST DATA RESULT
1717  004560  001401                BEQ     TS60
1718  004562  104000                EMT                     ;DATA RESULT OF NEGB INCORRECT
1719                               ;*****************************************************************************
1720                               ;TEST 60        TEST MODE 1 USING NEGATE INST.
1721                               ;*****************************************************************************
1722  004564                      TS60:
1723  004564  005000                CLR     R0              ;POINT TO LOC. 0
1724  004566  005010                CLR     (R0)            ;CLEAR LOC. 0
1725  004570  005210                INC     (R0)            ;LOC. 0=1
1726  004572  005410                NEG     (R0)            ;TRY NEG. LOC. 0=-1
1727  004574  100003                BPL     NEG10           ;CC=1001
1728  004576  001402                BEQ     NEG10
1729  004600  102401                BVS     NEG10
```

```
1730  004602  103401                  BCS     NEG11
1731  004604                  NEG10:
1732  004604  104000                  EMT                     ;NEGATE DID NOT SET CC'S CORRECTLY
1733
1734  004606  005237  000000  NEG11:  INC     @#0             ;TEST DATA RESULT
1735  004612  001401                  BEQ     NEG12
1736  004614  104000                  EMT                     ;DATA RESULT OF NEGATE INCORRECT
1737  004616  105110          NEG12:  COMB    (R0)            ;LOC. 0=377
1738  004620  105410                  NEGB    (R0)            ;TRY NEGB LOC. 0=1
1739  004622  100403                  BMI     NEG13           ;CC=0001?
1740  004624  001402                  BEQ     NEG13
1741  004626  102401                  BVS     NEG13
1742  004630  103401                  BCS     NEG14
1743  004632                  NEG13:
1744  004632  104000                  EMT                     ;NEGB DID NOT SET CC'S CORRECTLY
1745  004634  005337  000000  NEG14:  DEC     @#0             ;TEST DATA RESULT
1746  004640  001401                  BEQ     TS61
1747  004642  104000                  EMT                     ;DATA RESULT OF NEGB INCORRECT
1748                          ;*******************************************************************************
1749                          ;TEST 61        TEST MODE 2 USING NEGATE INSTRUCTION
1750                          ;*******************************************************************************
1751  004644                  TS61:
1752  004644  005000                  CLR     R0              ;POINT TO LOC. 0
1753  004646  005010                  CLR     (R0)            ;CLEAR LOC. 0
1754  004650  005210                  INC     (R0)            ;LOC. 0=1
1755  004652  005420                  NEG     (R0)+           ;TRY NEG.: LOC. 0=-1
1756  004654  100003                  BPL     NEG20           ;CC=1001?
1757  004656  001402                  BEQ     NEG20
1758  004660  102401                  BVS     NEG20
1759  004662  103401                  BCS     NEG21
1760  004664                  NEG20:
1761  004664  104000                  EMT                     ;NEGATE DID NOT SET CC'S CORRECTLY
1762  004666  105300          NEG21:  DECB    R0              ;R0=LOC. 0
1763  004670  105300                  DECB    R0
1764  004672  105420                  NEGB    (R0)+           ;BYTE 0=1   R0=1
1765  004674  105420                  NEGB    (R0)+           ;BYTE 1=1   R0=2
1766  004676  105340                  DECB    -(R0)           ;R0=1  LOC. 0=01
1767  004700  005300                  DEC     R0              ;R0=0
1768  004702  001401                  BEQ     NEG22
1769  004704  104000                  EMT                     ;REGISTER NOT INCREMENTED CORRECTLY
1770  004706  005337  000000  NEG22:  DEC     @#0             ;LOC. 0=0
1771  004712  001401                  BEQ     TS62
1772  004714  104000                  EMT                     ;NEG BYTE INSTRUCTIONS FAILED
1773
1774                          ;*******************************************************************************
1775                          ;
1776                          ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND INSTRUCTIONS.  IT
1777                          ;USES LOCATION 0 AS ITS TARGET DATA.  A TABLE LOCATED AT LOC. 400
1778                          ;THRU 402 IS USED TO SUPPLY THE ADDRESS OF LOCATION 0 TO THE
1779                          ;INSTRUCTIONS UNDER TEST.
1780                          ;       R0 IS SET TO 400, THE START OF THE ADDRESS TABLE, AND A CLR
1781                          ;INSTRUCTION IS EXECUTED WITH MODE 3 TO CLEAR LOC. 0.  THEN R0
1782                          ;IS DECREMENTED BY TWO AND TWO OTHER MODE 3 INSTRUCTIONS OPERATE ON
1783                          ;LOC. 0 TO VERIFY THE DATA RESULTS OF THE TEST.  THE PROPER INCREMENTING
1784                          ;OF THE REGISTER IS ALSO VERIFIED IN THIS MANNER.
1785                          ;       IF A FAILURE IS DETECTED BE SURE TO VERIFY THAT THE TABLE
```

```
1786
1787
1788
1789
1790
1791   004716
1792   004716   005000          ;(LOC. 400-402) HAS THE PROPER VALUES (0).
1793   004720   105100          ;
1794   004722   005200          ;*****************************************************************************
1795   004724   005010          ;TEST 62        TEST MODE 3 USING SOP INST.
1796   004726   005030          ;*****************************************************************************
1797   004730   001401  TS62:
1798   004732   104000          CLR     R0              ;SET R0=400
1799   004734   005300          COMB    R0
1800   004736   005300          INC     R0
1801   004740   005130          CLR     (R0)            ;CLEAR LOC 400
1802   004742   100002          CLR     @(R0)+          ;TRY TO CLEAR LOC 0 USING MODE 3 ;R0=402
1803   004744   005230          BEQ     SOP3A
1804   004746   001401          EMT                     ;CLR DID NOT SET Z-BIT
1805   C04750          SOP3A:  DEC     R0              ;RESET R0=400
1806   004750   104000          DEC     R0
1807                           COM     @(R0)+          ;TRY TO COMPLEMENT LOC 0 OF MODE 3  ;R0=402
1808                           BPL     SOP3B
1809                           INC     @(R0)+          ;TRY TO INCREMENT LOC 0 W/MODE 3  ;R0=404
1810                           BEQ     TS63
1811                   SOP3B:
1812                           EMT                     ;CUMMULATIVE RESULT OF ABOVE INST FAILED
1813
1814                   ;*****************************************************************************
1815                   ;
1816                   ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND BYTE INSTRUCTIONS
1817                   ;WHICH ADDRESS EVEN BYTES.  AGAIN, THE TARGET LOCATION 0 IS USED
1818                   ;AND THE SAME TABLE AT 400 IS EMPLOYED.
1819                   ;       AFTER POINTING R4 TO THE TABLE (400) AND SETTING LOCATION
1820                   ;0 TO -1, A CLRB INSTRUCTION IS USED TO CLEAR BYTE 0.
1821                   ;       SEVERAL OTHER MODE 3 INSTRUCTIONS ARE THEN USED WITH THE TABLE
1822                   ;TO VERIFY THE DATA RESULTS AND THE PROPER INCREMENTING OF THE REGISTER.
1823   004752          ;IF A FAILURE IS DETECTED, BE SURE THAT THE TABLE (LOCATION 400-402) HAS
1824   004752   005004 ;THE PROPER VALUES (0).
1825   004754   105104 ;
1826   004756   005204 ;*****************************************************************************
1827   004760   005000 ;TEST 63        TEST MODE 3 EVEN BYTE USING SOP INST.
1828   004762   005010 ;*****************************************************************************
1829   004764   005110 TS63:
1830   004766   105034          CLR     R4              ;SET R4=400
1831   004770   001401          COMB    R4
1832   004772   104000          INC     R4
1833   004774   005304          CLR     R0              ;INITIALIZE LOC. 0=-1
1834   004776   005304          CLR     (R0)
1835   005000   005234          COM     (R0)            ;LOC. 0=-1
1836   005002   100006          CLRB    @(R4)+          ;TRY TO CLEAR EVEN BYTE ;LOC. 0=177400  R4=402
1837   005004   105434          BEQ     SOPB3A
1838   005006   100004          EMT                     ;CLRB DID NOT SET Z-BIT
1839   005010   005304  SOPB3A: DEC     R4              ;RESET POINTER  R4=400
1840   005012   005304          DEC     R4
1841   005014   105234          INC     @(R4)+          ;TRY INCREMENTING WORD  LOC.0=177401  R4=402
                               BPL     SOPB3B
                               NEGB    @(R4)+          ;TRY TO NEGATE EVEN BYTE ;LOC.0=-1 R4=404
                               BPL     SOPB3B
                               DEC     R4              ;R4=402
                               DEC     R4
                               INCB    @(R4)+          ;TRY TO INCREMENT EVEN BYTE ;LOC. 0=17400
```

```
1842  005016  001401          BEQ     TS64
1843  005020
1844  005020  104000  SOPB3B: EMT                     ;CUMMULATIVE RESULT OF ABOVE INST FAILED
1845
1846  ;*****************************************************************************************
1847  ;
1848  ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND BYTE INSTRUCTIONS
1849  ;WHICH ADDRESS ODD BYTES.  THE TARGET IS BYTE 1.  A TABLE AT
1850  ;LOC. 400-406 IS USED.  R0 SERVES AS THE TABLE POINTER.
1851  ;       R0 IS INITIALIZED TO 400.  LOC. 0 IS SET TO -1 USING THE
1852  ;FIRST TWO TABLE ENTRIES.  A CLRB MODE 3 IS EXECUTED ON BYTE 1 USING
1853  ;TABLE ADDRESS AT 404.  R0 IS DECREMENTED TO 402 AND SEVERAL SOP
1854  ;MODE 3 INSTRUCTIONS ARE USED TO VERIFY DATA RESULTS AND PROPER
1855  ;REGISTER INCREMENTING.
1856  ;       THE TABLE (400-406) SHOULD CONTAIN 0,0,1,1 BEFORE AND
1857  ;AFTER THE TEST IS RUN.
1858  ;
```

```
1859
1860
1861
1862  005022
1863  005022  005000
1864  005024  105100
1865  005026  005200
1866  005030  005030
1867  005032  005130
1868  005034  105030
1869  005036  001401
1870  005040  104000
1871  005042  005300
1872  005044  005300
1873  005046  005300
1874  005050  005300
1875  005052  005230
```

```
;*****************************************************************************************
;TEST 64         TEST MODE 3 ODD BYTE USING SOP INST.
;*****************************************************************************************
TS64:
        CLR     R0              ;SET R0=400
        COMB    R0
        INC     R0
        CLR     @(R0)+          ;INITIALIZE
        COM     @(R0)+          ;LOC 0=-1 R0=404
        CLRB    @(R0)+          ;TRY TO CLEAR ODD BYTE LOC. 0=377 R0=406
        BEQ     SOPB3C
        EMT                     ;CLRB DID NOT SET Z-BIT
SOPB3C: DEC     R0              ;RESET R0=402
        DEC     R0
        DEC     R0              ;POINT TO EVEN BYTE ADDR.
        DEC     R0
        INC     @(R0)+          ;INCREMENT WORD LOC. 0=400 R0=404
```

```
1876  005054  105430
1877  005056  100002
1878  005060  105230
1879  005062  001401
1880  005064
1881  005064  104000
1882
1883
1884
1885  005066
1886  005066  005000
1887  005070  105100
1888  005072  005200
1889  005074  005010
1890  005076  005004
1891  005100  005014
1892  005102  005214
1893  005104  005430
1894  005106  100003
1895  005110  001402
1896  005112  102401
1897  005114  103401
1898  005116
1899  005116  104000
1900  005120  005214
1901  005122  001401
1902  005124  104000
1903  005126  105137  000001
1904  005132  005237  000000
1905  005136  105430
1906  005140  100401
1907  005142  104000
1908  005144  105430
1909  005146  100001
1910  005150  104000
1911  005152  105137  000001
1912  005156  105237  000001
1913  005162  005214
1914  005164  001401
1915  005166  104000
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928  005170
1929  005170  005000
1930  005172  105100
1931  005174  005200
```

```
        NEGB    @(R0)+          ;TRY TO NEGATE ODD BYTE LOC. 0=177400 R0=406
        BPL     SOPB3D
        INCB    @(R0)+          ;TRY TO INCREMENT ODD BYTE LOC.0=0 R0=410
        BEQ     TS65
SOPB3D:
        EMT                     ;CUMMULATIVE RESULT OF ABOVE INSTS FAILED
;*****************************************************************************
;TEST 65         TEST MODE 3 USING NEGATE INSTRUCTION
;*****************************************************************************
TS65:
        CLR     R0              ;R0=400
        COMB    R0
        INC     R0
        CLR     (R0)            ;LOC. 400=0
        CLR     R4              ;R4=0
        CLR     (R4)            ;LOC. 0=0
        INC     (R4)            ;LOC. 0=1
        NEG     @(R0)+          ;TRY NEGATE   LOC. 0=-1  R0=402
        BPL     NEG30           ;CC=1001?
        BEQ     NEG30
        BVS     NEG30
        BCS     NEG31
NEG30:
        EMT                     ;NEG DID NOT SET CC'S CORRECTLY
NEG31:  INC     (R4)            ;LOC. 0=0
        BEQ     NEG32
        EMT                     ;DATA RESULT OF NEG INCORRECT
NEG32:  COMB    @#1             ;LOC 0=177400
        INC     @#0             ;LOC. 0=177401
        NEGB    @(R0)+          ;TRY NEGB LOC. 0=177777  R0=404
        BMI     NEG33
        EMT                     ;NEGB FAILED WITH EVEN BYTE
NEG33:  NEGB    @(R0)+          ;TRY NEGB  LOC.0=777  R0=406
        BPL     NEG34
        EMT                     ;NEGB FAILED WITH ODD BYTE
NEG34:  COMB    @#1             ;LOC. 0=177377
        INCB    @#1             ;LOC. 0=177777
        INC     (R4)            ;LOC. 0=0
        BEQ     TS66
        EMT                     ;DATA RESULT OF NEGB'S INCORRECT

;*****************************************************************************
;
;       THIS TEST VERIFIES MODE 4 SINGLE OPERAND INSTRUCTIONS.
;R0 IS SET TO 400. A CLR INSTRUCTION IS EXECUTED IN MODE 4 TO CLEAR
;LOC. 376. R0 IS RESET TO 400 AND A COM INSTRUCTION USING MODE 4
;COMPLEMENTS LOC.376.
;       TWO INC INSTRUCTIONS AND A MODE 4 INSTRUCTION ARE EXECUTED
;TO COMPLETE THE TEST.
;*****************************************************************************
;TEST 66         TEST MODE 4 USING SOP INSTS
;*****************************************************************************
TS66:
        CLR     R0              ;SET R0=400
        COMB    R0
        INC     R0
```

```
1932  005176  005040          CLR     -(R0)           ;TRY TO CLEAR USING MODE 4
1933  005200  001401          BEQ     SOP4A
1934  005202  104000          EMT                     ;CLR DID NOT SET Z-BIT
1935  005204  005200  SOP4A:  INC     R0              ;RESET R0
1936  005206  005200          INC     R0
1937  005210  005140          COM     -(R0)           ;TRY TO COMPLEMENT USING MODE 4
1938  005212  100004          BPL     SOP4B
1939  005214  005200          INC     R0              ;MOVE POINTER
1940  005216  005200          INC     R0
1941  005220  005240          INC     -(R0)
1942  005222  001401          BEQ     TS67
1943  005224          SOP4B:
1944  005224  104000          EMT                     ;CHECK CUMMULATIVE RESULT OF ABOVE INST.
1945
1946                  ;*****************************************************************************
1947                  ;
1948                  ;       THIS TEST VERIFIES MODE 5 SINGLE OPERAND INSTRUCTIONS.  IT
1949                  ;USES LOCATION 0 AS ITS TARGET DATA.  A TABLE LOCATED AT LOC. 372
1950                  ;THRU 374 IS USED TO SUPPLY THE ADDRESS OF LOCATION 0 TO THE
1951                  ;INSTRUCTIONS UNDER TEST.
1952                  ;       R0 IS SET TO 376, (THE START OF THE ADDRESS TABLE) +2,
1953                  ;AND A CLR INSTRUCTION IS EXECUTED WITH MODE 3 TO CLEAR
1954                  ;LOC. 0.  THEN R0 IS INCREMENTED BY TWO AND TWO OTHER MODE 3
1955                  ;INSTRUCTIONS OPERATE ON LOC. 0 TO VERIFY THE DATA RESULTS OF
1956                  ;THE TEST.  THE PROPER DECREMENTING OF THE REGISTER IS ALSO
1957                  ;VERIFIED IN THIS MANNER.
1958                  ;       IF A FAILURE IS DETECTED BE SURE TO VERIFY THAT THE TABLE
1959                  ;(LOC. 372 THRU 374) HAS THE PROPER VALUES (0).
1960                  ;
1961                  ;*****************************************************************************
1962                  ;TEST 67        TEST MODE 5 USING SOP INSTS
1963                  ;*****************************************************************************
1964  005226  TS67:
1965  005226  012700  000370          MOV     #370,R0         ;CLEAR LOCATION 370-376
1966  005232  005020                  CLR     (R0)+           ;370
1967  005234  005020                  CLR     (R0)+           ;372
1968  005236  005020                  CLR     (R0)+           ;374
1969  005240  005010                  CLR     (R0)            ;376
1970  005242  005000                  CLR     R0              ;SET R0=376 (LOW BYTE)
1971  005244  005020                  CLR     (R0)+
1972  005246  105400                  NEGB    R0
1973  005250  005050                  CLR     @-(R0)          ;TRY TO CLEAR LOC 0 W/MODE 5
1974  005252  001401                  BEQ     SOP5A
1975  005254  104000                  EMT                     ;CLR DID NOT SET Z-BIT
1976  005256  005200          SOP5A:  INC     R0              ;RESET R0
1977  005260  005200                  INC     R0
1978  005262  005150                  COM     @-(R0)          ;TRY TO COMPLEMENT LOC. 0 W/MODE 5
1979  005264  100002                  BPL     SOP5B
1980  005266  005250                  INC     @-(R0)          ;TRY TO INCREMENT LOC. 0 W/MODE 5
1981  005270  001401                  BEQ     TS70
1982  005272          SOP5B:
1983  005272  104000                  EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INSTS
1984
1985                  ;*****************************************************************************
1986                  ;
1987                  ;       THIS TEST VERIFIES MODE 6 SINGLE OPERAND INSTRUCTIONS.  IT
```

```
1988
1989
1990
1991
1992
1993
1994
1995
1996  005274
1997  005274  005000                ;USES LOCATION 0 AS ITS TARGET DATA.  R0 IS SET TO 400 USING
1998  005276  105100                ;PREVIOUSLY TESTED INSTRUCTIONS AND A MODE 6 CLR INSTRUCTION IS
1999  005300  005200                ;EXECUTED ON LOC. 0 USING R0 AND A -400 OFFSET.  COM AND INC
2000  005302  005060  177400        ;INSTRUCTIONS ARE THEN USED TO VERIFY THE DATA.
2001  005306  001401                ;
2002  005310  104000                ;*****************************************************************************
2003  005312  005160  177400        ;TEST 70        TEST MODE 6 USING SOP INSTS
2004  005316  100003                ;*****************************************************************************
2005  005320  005260  177400        TS70:
2006  005324  001401                        CLR     R0              ;SET R0=400
2007  005326                                COMB    R0
2008  005326  104000                        INC     R0
2009                                  SOP6A:  CLR     -400(R0)        ;TRY TO CLEAR LOCATION 0 W/MODE 6
2010                                          BEQ     SOP6A
2011                                          EMT                     ;CLR DID NOT SET Z-BIT
2012                                          COM     -400(R0)        ;TRY TO COMPLEMENT LOCATION 0 W/MODE 6
2013                                          BPL     SOP6B
2014                                          INC     -400(R0)        ;TRY TO INCREMENT LOCATION 0 W/MODE 6
2015                                          BEQ     TS71
2016                                  SOP6B:
2017                                          EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INSTS
2018
2019                                  ;*****************************************************************************
2020                                  ;
2021                                  ;       THIS TEST VERIFIES MODE 7 SINGLE OPERAND INSTRUCTIONS.  IT USES
2022  005330                          ;THE POINTER TO LOC. 0 WHICH IS STORED AT LOC. 402.
2023  005330  005000                ;       R0 IS SET TO 400 AND A MODE 7 CLR INSTRUCTION IS
2024  005332  105100                ;EXECUTED WITH A +2 OFFSET TO CLEAR LOC. 0.
2025  005334  005200                ;       SEVERAL OTHER MODE 7 INSTRUCTIONS ARE THEN USED ON THE COMMON
2026  005336  005210                ;LOCATION TO VERIFY THE DATA RESULTS.
2027  005340  005070  000002        ;
2028  005344  001401                ;*****************************************************************************
2029  005346  104000                ;TEST 71        TEST MODE 7 USING SOP INST.
2030  005350  005170  000002        ;*****************************************************************************
2031  005354  100003                TS71:
2032  005356  005270  000002                CLR     R0              ;SET R0=400
2033  005362  001401                        COMB    R0
2034  005364                                INC     R0
2035  005364  104000                        INC     (R0)            ;R0=1
2036                                        CLR     @2(R0)          ;TRY TO CLEAR LOC. 0 W/MODE 7
2037                                        BEQ     SOP7A
2038                                        EMT                     ;CLR DID NOT SET Z-BIT
2039                                  SOP7A:  COM     @2(R0)          ;TRY TO COMPLEMENT LOC. 0 W/MODE 7
2040  005366                                BPL     SOP7B
2041  005366  005000                        INC     @2(R0)          ;TRY TO INCREMENT LOC. 0 W/MODE 7
2042  005370  005010                        BEQ     TS72
2043  005372  005120                SOP7B:
                                        EMT                     ;TEST CUMMULATIVE RESULT OF ABOVE INSTS.

                                ;*****************************************************************************
                                ;TEST 72        TEST MODE 4 WITH NEGATE INSTRUCTION
                                ;*****************************************************************************
                                TS72:
                                        CLR     R0
                                        CLR     (R0)
                                        COM     (R0)+           ;LOC. 0=177777,  R0=2
```

```
2044  005374  005440                        NEG     -(R0)           ;TRY NEGATE, LOC. 0=1
2045  005376  100403                        BMI     NEG40           ;CC=0001?
2046  005400  001402                        BEQ     NEG40
2047  005402  102401                        BVS     NEG40
2048  005404  103401                        BCS     NEG41
2049  005406                          NEG40:
2050  005406  104000                        EMT                     ;NEG DID NOT SET CC'S CORRECTLY
2051  005410  005400                  NEG41: NEG     R0              ;TST R0 WITH A NEG.
2052  005412  001401                        BEQ     NEG42
2053  005414  104000                        EMT                     ;R0 NOT DECREMENTED PROPERLY
2054  005416  005310                  NEG42: DEC     (R0)            ;TEST DTA RESULT OF NEG
2055  005420  001401                        BEQ     TS73
2056  005422  104000                        EMT                     ;DATA RESULT OF NEG INCORRECT
2057                                  ;*****************************************************************************
2058                                  ;TEST 73        TEST MODE 5 WITH NEGATE INSTRUCTION
2059                                  ;*****************************************************************************
2060  005424                          TS73:
2061  005424  005000                        CLR     R0              ;R0=0
2062  005426  005010                        CLR     (R0)            ;LOC. 0=0
2063  005430  105100                        COMB    R0              ;R0=377
2064  005432  005200                        INC     R0              ;R0=400
2065  005434  005010                        CLR     (R0)            ;SET 400 = 0
2066  005436  005004                        CLR     R4              ;R4=0
2067  005440  005314                        DEC     (R4)            ;LOC. 0=177777
2068  005442  005450                        NEG     @-(R0)          ;TRY NEGATE:  LOC. 0=1
2069  005444  100403                        BMI     NEG50           ;CC=0001?
2070  005446  001402                        BEQ     NEG50
2071  005450  102401                        BVS     NEG50
2072  005452  103401                        BCS     NEG51
2073  005454                          NEG50:
2074  005454  104000                        EMT                     ;NEG DID NOT SET CC'S CORRECTLY
2075  005456  005314                  NEG51: DEC     (R4)
2076  005460  001401                        BEQ     NEG52
2077  005462  104000                        EMT                     ;DATA RESULT OF NEG INCORRECT
2078  005464  105100                  NEG52: COMB    R0
2079  005466  005300                        DEC     R0
2080  005470  001401                        BEQ     TS74
2081  005472  104000                        EMT                     ;REGISTER NOT DECREMENTED PROPERLY
2082                                  ;*****************************************************************************
2083                                  ;TEST 74        TEST MODE 6 WITH NEGATE
2084                                  ;*****************************************************************************
2085  005474                          TS74:
2086  005474  005000                        CLR     R0              ;R0=0
2087  005476  005004                        CLR     R4              ;R4=0
2088  005500  105100                        COMB    R0              ;R0=377
2089  005502  005014                        CLR     (R4)            ;LOC. 0=0
2090  005504  105024                        CLRB    (R4)+           ;LOC. 0=177777,  R4=1
2091  005506  105114                        COMB    (R4)            ;LOC. 0=177400
2092  005510  005460  177401               NEG     -377(R0)        ;LOC. 0=400
2093  005514  100403                        BMI     NEG60           ;CC=0001
2094  005516  001402                        BEQ     NEG60
2095  005520  102401                        BVS     NEG60
2096  005522  103401                        BCS     NEG61
2097  005524                          NEG60:
2098  005524  104000                        EMT                     ;NEG DID NOT SET CC'S CORRECTLY
2099  005526  105314                  NEG61: DECB    (R4)
```

```
2100  005530  001401                        BEQ     TS75
2101  005532  104000                        EMT                     ;DATA RESULT OF NEG INCORRECT
2102                                ;*****************************************************************************
2103                                ;TEST 75        TEST MODE 7 W/ NEGATE
2104                                ;*****************************************************************************
2105  005534                        TS75:
2106  005534  005000                        CLR     R0              ;R0=0
2107  005536  005010                        CLR     (R0)            ;LOC. 0=0
2108  005540  005110                        COM     (R0)            ;LOC. 0=177777
2109  005542  105100                        COMB    R0              ;R0=377
2110  005544  105470  000005                NEGB    @5(R0)          ;R0+5=404,  404=1,  LOC. 0=777
2111  005550  100403                        BMI     NEG70           ;CC=0001?
2112  005552  001402                        BEQ     NEG70
2113  005554  102401                        BVS     NEG70
2114  005556  103401                        BCS     NEG71
2115  005560                        NEG70:
2116  005560  104000                        EMT                     ;NEG DID NOT SET CC'S CORRECTLY
2117  005562  105100                NEG71:  COMB    R0              ;R0=0
2118  005564  105120                        COMB    (R0)+           ;LOC. 0=400,  R0=1
2119  005566  105310                        DECB    (R0)            ;LOC. 0=0
2120  005570  005467  172204                NEG     0               ;USE NEG MODE 67 TO TST FOR ZERO
2121  005574  001401                        BEQ     TS76
2122  005576  104000                        EMT                     ;DATA RESULT OF NEG WAS INCORRECT
2123                                ;*****************************************************************************
2124                                ;
2125                                ;       THIS TEST VERIFIES PROGRAM COUNTER ADDRESSING WITH SOP
2126                                ;INSTRUCTIONS.  CLR MODE 77 IS USED TO CLEAR THE LOCATION FOLLOWING THE
2127                                ;INSTRUCTION (SOPX).  THEN SINGLE OPERAND INSTRUCTIONS WITH MODES 37, 67, AND
2128                                ;77, USING INDIRECT POINTER SOPXAD ARE USED TO VERIFY THE DATA RESULTS
2129                                ;OF THESE INSTRUCTIONS.
2130                                ;
2131                                ;*****************************************************************************
2132                                ;TEST 76        TEST SOP INSTRUCTIONS MODES 2,3,6,7 WITH REGISTER 7
2133                                ;*****************************************************************************
2134                                TS76:
2135  005600                                CLR     (R7)+           ;CLEAR NEXT LOCATION: (SOPX)
2136  005600  005027                SOPX:   -1                      ;USE MODE 27
2137  005602  177777                        BEQ     SOPA
2138  005604  001401                        EMT                     ;CLR DID NOT SET Z-BIT
2139  005606  104000                SOPA:   INC     @#SOPX          ;INC SOPX W/MODE 37
2140  005610  005237  005602                NEG     SOPX            ;NEGATE SOPX W/MODE 67
2141  005614  005467  177762                BPL     SOPB
2142  005620  100003                        INC     @SOPXAD         ;INC SOPX W/MODE 77
2143  005622  005277  000004                BEQ     TS77
2144  005626  001402                SOPB:
2145  005630                                EMT                     ;INC DID NOT SET Z-BIT
2146  005630  104000                SOPXAD: SOPX                    ;INDIRECT ADDRESS OF SOPX
2147  005632  005602                ;*****************************************************************************
2148                                ;
2149                                ;       THIS TEST VERIFIES SINGLE OPERAND NON-MODIFYING INSTRUCTIONS
2150                                ;USING MODE 0.  R0 IS SET TO ZERO AND THE CONDITION CODES ARE SET
2151                                ;TO THE COMPLEMENT OF THAT EXPECTED BY THE INSTRUCTION.  A TST INSTRUCTION
2152                                ;IS EXECUTED AND CONDITIONAL BRANCHES ARE USED TO TEST THE CONDITION
2153                                ;CODES.
2154
2155
```

```
005634
005634  005000
005636  000277
005640  000244
005642  005700
005644  102403
005646  100402
005650  103401
005652  001401
005654
005654  104000


005656
005656  005000
005660  105100
005662  000277
005664  000250
005666  105700
005670  102402
005672  101401
005674  100401
005676
005676  104000


005700
005700  005000
005702  005010
005704  000277
005706  000244
005710  005710
```

```
;
;*****************************************************************************
;TEST 77         TEST MODE 0 SOP NON-MODIFYING
;*****************************************************************************
TS77:
        CLR     R0              ;INITIALIZE R0=0
        SCC                     ;SET CC=1011
        CLZ
        TST     R0              ;TRY TST W/ MODE 0
        BVS     SNM0A           ;CHECK THAT CC=0100
        BMI     SNM0A
        BCS     SNM0A
        BEQ     TS100
SNM0A:
        EMT                     ;CONDITION CODES NOT SET PROPERLY

;*****************************************************************************
;
;       THIS TEST VERIFIES SINGLE OPERAND NON-MODIFYING BYTE INSTRUCTIONS WITH MODE 0.
;R0 IS SET TO 377 AND COMPLEMENT OF THE EXPECTED CONDITION CODES
;IS LOADED IN PSW.  A TSTB INSTRUCTION IS EXECUTED AND THE RESULTS
;ARE CHECKED WITH SEVERAL CONDITIONAL BRANCH INSTRUCTIONS.
;       THIS VERIFIES THAT THE PROPER BYTE WAS TESTED.
;
;*****************************************************************************
;TEST 100        TEST MODE 0 EVEN BYTE W/ SOP NON-MODIFYING
;*****************************************************************************
TS100:
        CLR     R0              ;INITIALIZE
        COMB    R0              ;R0=377
        SCC                     ;SET CC=0111
        CLN
        TSTB    R0              ;TRY TST EVEN BYTE
        BVS     SNMB0A          ;CHECK CC=1000
        BLOS    SNMB0A
        BMI     TS101
SNMB0A:
        EMT                     ;CONDITION CODES NOT SET PROPERLY

;*****************************************************************************
;
;       THIS TEST VERIFIES SINGLE OPERAND INSTRUCTIONS WITH MODE 1.
;R0 IS USED TO POINT TO AND CLEAR LOC. 0.  THE COMPLEMENT OF THE
;EXPECTED CONDITION CODES ARE LOADED IN THE PSW.  A TST INSTRUCTION
;IS THEN EXECUTED ON LOC. 0 USING R0 AND CONDITIONAL BRANCHES TEST
;THE RESULTS.
;
;*****************************************************************************
;TEST 101        TEST MODE 1 SOP NON-MODIFYING
;*****************************************************************************
TS101:
        CLR     R0              ;POINT TO LOC 0
        CLR     (R0)            ;CLEAR LOC 0
        SCC                     ;INITIALIZE
        CLZ                     ;CC=1011
        TST     (R0)            ;TRY TST W/ MODE 1
```

```
2212  005712  102403          BVS     SNM1A           ;CHECK CC=0100
2213  005714  103402          BCS     SNM1A
2214  005716  100401          BMI     SNM1A
2215  005720  001401          BEQ     TS102
2216  005722                SNM1A:
2217  005722  104000          EMT                     ;CC'S NOT SET PROPERLY
2218
2219                          ;*****************************************************************************
2220                          ;
2221                          ;       THIS TEST SETS LOCATION 0 TO 377 AND THEN USES R0 TO TEST
2222                          ;THE EVEN BYTE AND THE ODD BYTE USING SOP BYTE INSTRUCTIONS WITH MODE 1.
2223                          ;AGAIN, CONDITIONAL BRANCHES ARE USED TO VERIFY THE SETTING OF THE
2224                          ;PROPER CONDITION CODE BITS.
2225                          ;
2226                          ;*****************************************************************************
2227                          ;TEST 102       TEST MODE 1 BYTE INST. NON-MODIFYING
2228                          ;*****************************************************************************
2229  005724                TS102:
2230  005724  005000          CLR     R0              ;POINT TO LOC 0
2231  005726  005010          CLR     (R0)            ;CLEAR LOC 0
2232  005730  105110          COMB    (R0)            ;COMPLEMENT BYTE 0
2233  005732  000277          SCC                     ;SET CC=0111
2234  005734  000250          CLN
2235  005736  105710          TSTB    (R0)            ;TRY TST ON EVEN BYTE
2236  005740  102402          BVS     SNMB1A
2237  005742  101401          BLOS    SNMB1A
2238  005744  100401          BMI     SNMB1B
2239  005746                SNMB1A:
2240  005746  104000          EMT                     ;CC'S NOT CORRECT
2241  005750  005000        SNMB1B: CLR     R0
2242  005752  005200          INC     R0
2243  005754  000277          SCC                     ;SET CC=1011
2244  005756  000244          CLZ
2245  005760  105710          TSTB    (R0)            ;TRY TO TST AN ODD BYTE
2246  005762  102403          BVS     SNMB1C          ;CHECK CC=0100
2247  005764  103402          BCS     SNMB1C
2248  005766  100401          BMI     SNMB1C
2249  005770  001401          BEQ     TS103
2250  005772                SNMB1C:
2251  005772  104000          EMT                     ;CC'S NOT CORRECT
2252
2253                          ;*****************************************************************************
2254                          ;
2255                          ;       THIS TEST VERIFIES THE SINGLE-OPERAND NON-MODIFYING INSTRUCTIONS
2256                          ;USING MODE 2.  IT USES THE IDENTICAL PROCEDURE EMPLOYED IN THE
2257                          ;MODE 1 TESTS.  ADDITIONALLY, THE REGISTER IS CHECKED TO ASSURE THAT
2258                          ;IT IS INCREMENTED PROPERLY.
2259                          ;
2260                          ;*****************************************************************************
2261                          ;TEST 103       TEST MODE 2 WITH SOP NON-MODIFYING
2262                          ;*****************************************************************************
2263  005774                TS103:
2264  005774  005000          CLR     R0              ;INITIALIZE R0=0
2265  005776  005010          CLR     (R0)            ;CLEAR LOC 0
2266  006000  000277          SCC                     ;SET CC=1011
2267  006002  000244          CLZ
```

```
2268  006004  005720          TST     (R0)+           ;TRY TST W/ MODE 2
2269  006006  102403          BVS     SNM2A           ;CHECK CC=0100
2270  006010  103402          BCS     SNM2A
2271  006012  100401          BMI     SNM2A
2272  006014  001401          BEQ     SNM2B
2273  006016                SNM2A:
2274  006016  104000          EMT                     ;CC'S NOT CORRECT
2275  006020  005300  SNM2B:  DEC     R0              ;RESET R0
2276  006022  005300          DEC     R0
2277  006024  001401          BEQ     TS104
2278  006026  104000          EMT                     ;MODE 2 DID NOT INC REQ CORRECTLY
2279
2280                ;*****************************************************************************
2281                ;
2282                ;       THIS TEST VERIFIES MODE 2 SINGLE OPERAND NON-MODIFYING BYTE
2283                ;INSTRUCTIONS IT USES R0 TO POINT TO LOC. 0.  WITH LOCATION 0
2284                ;SET TO 377, THE EVEN AND ODD BYTE IS TESTED WITH TSTB INSTRUCTIONS
2285                ;TO VERIFY THE CORRECT CC ARE SET.  THE REGISTER IS CHECKED FOR
2286                ;PROPER INCREMENTING.
2287                ;
2288                ;*****************************************************************************
2289                ;TEST 104       TEST MODE 2 - BYTE W/ SOP NON-MODIFYING
2290                ;*****************************************************************************
2291  006030        TS104:
2292  006030  005000          CLR     R0              ;CLEAR R0
2293  006032  005010          CLR     (R0)            ;CLEAR LOC 0
2294  006034  105110          COMB    (R0)            ;SET LOC 0=377
2295  006036  000277          SCC                     ;SET CC=0111
2296  006040  000250          CLN
2297  006042  105720          TSTB    (R0)+           ;TRY TST OF EVEN BYTE
2298  006044  102402          BVS     SNMB2A
2299  006046  101401          BLOS    SNMB2A
2300  006050  100401          BMI     SNMB2B
2301  006052                SNMB2A:
2302  006052  104000          EMT                     ;CC'S NOT SET CORRECTLY
2303  006054  005300  SNMB2B: DEC     R0              ;DECREMENT R0
2304  006056  001401          BEQ     SNMB2C
2305  006060  104000          EMT                     ;MODE 2 DID NOT INC REG CORRECTLY
2306  006062  005200  SNMB2C: INC     R0              ;POINT TO ODD BYTE
2307  006064  000277          SCC                     ;SET CC=1011
2308  006066  000244          CLZ
2309  006070  105720          TSTB    (R0)+           ;TRY TST OF ODD BYTE
2310  006072  102403          BVS     SNMB2D          ;CHECK CC'S=0100
2311  006074  103402          BCS     SNMB2D
2312  006076  100401          BMI     SNMB2D
2313  006100  001401          BEQ     SNMB2E
2314  006102                SNMB2D:
2315  006102  104000          EMT                     ;CC'S NOT CORRECT
2316  006104  005300  SNMB2E: DEC     R0
2317  006106  005300          DEC     R0
2318  006110  001401          BEQ     TS105
2319  006112  104000          EMT                     ;R0 DID NOT INCREMENT PROPERLY
2320
2321                ;*****************************************************************************
2322                ;
2323                ;       THIS TEST VERIFIES MODE 3 SINGLE OPERAND NON-MODIFYING INSTRUCTIONS.
```

```
2324
2325
2326
2327
2328
2329
2330
2331  006114
2332  006114  005000
2333  006116  005010
2334  006120  105100
2335  006122  005300
2336  006124  000277
2337  006126  000244
2338  006130  005730
2339  006132  102403
2340  006134  103402
2341  006136  100401
2342  006140  001401
2343  006142
2344  006142  104000
2345  006144  005300
2346  006146  105100
2347  006150  001401
2348  006152  104000
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362  006154
2363  006154  005000
2364  006156  005010
2365  006160  105110
2366  006162  105100
2367  006164  005200
2368  006166  005720
2369  006170  000277
2370  006172  000250
2371  006174  105730
2372  006176  102402
2373  006200  101401
2374  006202  100401
2375  006204
2376  006204  104000
2377  006206  000277
2378  006210  000244
2379  006212  105730
```

```
;A POINTER IN A TABLE AT LOC. 376 IS USED TO TEST LOCATION 0.
;THE CC'S AND THE REGISTER ARE CHECKED FOLLOWING THE
;TST MODE 3 INSTRUCTION.
;
;*****************************************************************************
;TEST 105        TEST MODE 3 W/ SOP NON-MODIFYING INSTS
;*****************************************************************************
TS105:
        CLR     R0              ;R0=0
        CLR     (R0)            ;CLEAR LOC 0
        COMB    R0              ;R0=376
        DEC     R0
        SCC                     ;SET CC=1011
        CLZ
        TST     @(R0)+          ;TRY TST W/ MODE 3
        BVS     SNM3A           ;CHECK CC=0100
        BCS     SNM3A
        BMI     SNM3A
        BEQ     SNM3B
SNM3A:
        EMT                     ;CC'S NOT CORRECT
SNM3B:  DEC     R0              ;R0=377
        COMB    R0              ;R0=0
        BEQ     TS106
        EMT                     ;MODE 3 DID NOT INC REG CORRECTLY

;*****************************************************************************
;
;       THIS TEST VERIFIES SOP NON-MODIFYING BYTE INSTRUCTIONS MODE 3
;LOC. 0 IS SET TO 377.  TABLE AT LOC. 402-404 IS USED TO TEST
;BYTE 0 AND BYTE 1.  THE REGISTER IS CHECKED FOR PROPER INCREMENTING AND
;THE CC'S ARE VERIFIED.
;       THE TABLE AT LOC. 402-404 SHOULD CONTAIN 0 AND 1 BEFORE AND
;AFTER THE TEST IS RUN.
;
;*****************************************************************************
;TEST 106        TEST MODE 3 - BYTES W/ SOP NON-MODIFYING INSTS.
;*****************************************************************************
TS106:
        CLR     R0              ;R0=0
        CLR     (R0)            ;CLEAR LOC 0
        COMB    (R0)            ;LOC. 0 =377
        COMB    R0
        INC     R0
        TST     (R0)+           ;R0=402
        SCC                     ;CC=0111
        CLN
        TSTB    @(R0)+          ;TRY TST OF EVEN BYTE
        BVS     SNMB3A          ;CHECK CC=1000
        BLOS    SNMB3A
        BMI     SNMB3B
SNMB3A:
        EMT                     ;CC'S NOT CORRECT
SNMB3B: SCC                     ;SET CC=1011
        CLZ
        TSTB    @(R0)+          ;TRY TST OF ODD BYTE
```

```
2380  006214  102403                BVS     SNMB3C          ;CHECK CC=0100
2381  006216  103402                BCS     SNMB3C
2382  006220  100401                BMI     SNMB3C
2383  006222  001401                BEQ     SNMB3D
2384  006224                SNMB3C:
2385  006224  104000                EMT                     ;CC'S NOT CORRECT
2386  006226  005720        SNMB3D: TST     (R0)+           ;R0=410
2387  006230  005710                TST     (R0)
2388  006232  100401                BMI     TS107
2389  006234  104000                EMT                     ;TSTB DID NOT INCREMENT R0 CORRETCLY
2390                        ;*******************************************************************************
2391                        ;
2392                        ;       THIS TEST VERIFIES MODE 4 SOP NON-MODIFYING INSTRUCTIONS.
2393                        ;LOC. 0 IS SET TO -1 AND THE CC'S ARE SET TO THE COMPLEMENT OF THE
2394                        ;EXPECTED RESULTS.  R0 AND SET TO 2 AND A TST MODE 4 IS EXECUTED.
2395                        ;THE CC'S ARE CHECKED WITH CONDITIONAL BRANCH INSTRUCTIONS AND THE REGISTER
2396                        ;IS CHECKED FOR PROPER DECREMENTING.
2397                        ;
2398                        ;*******************************************************************************
2399                        ;TEST 107       TEST MODE 4 W/ SOP NON-MODIFYING INSTS
2400                        ;*******************************************************************************
2401  006236                TS107:
2402  006236  005000                CLR     R0              ;R0=0
2403  006240  005010                CLR     (R0)            ;LOC 0=0
2404  006242  005120                COM     (R0)+           ;LOC 0=-1
2405  006244  000277                SCC                     ;SET CC=1011
2406  006246  000244                CLZ
2407  006250  005740                TST     -(R0)           ;TRY TST W/ MODE 4
2408  006252  102402                BVS     SNM4A           ;CHECK CC=0100
2409  006254  101401                BLOS    SNM4A
2410  006256  100401                BMI     SNM4B
2411  006260                SNM4A:
2412  006260  104000                EMT                     ;CC'S NOT CORRECT
2413  006262  005700        SNM4B:  TST     R0
2414  006264  001401                BEQ     TS110
2415  006266  104000                EMT                     ;TST MODE 4 DID NOT DEC R0 CORRECTLY
2416
2417                        ;*******************************************************************************
2418                        ;
2419                        ;       THIS TEST VERIFIES MODE 5 SOP NON-MODIFYING INSTRUCTIONS.
2420                        ;IT USES A POINTER AT LOC. 376 TO TEST LOC. 0.  R0 IS SET
2421                        ;TO 400, A TST MODE 5 INSTRUCTION IS EXECUTED AND THE CC'S CHECKED.
2422                        ;R0 IS CHECKED TO INSURE PROPER DECREMENTING.
2423                        ;
2424                        ;*******************************************************************************
2425                        ;TEST 110       TEST MODE 5 W/ SOP NON-MODIFYING INSTS
2426                        ;*******************************************************************************
2427  006270                TS110:
2428  006270  005000                CLR     R0              ;R0=0
2429  006272  005010                CLR     (R0)            ;LOC 0=0
2430  006274  005110                COM     (R0)            ;LOC 0=-1
2431  006276  105100                COMB    R0              ;R0=377
2432  006300  005200                INC     R0              ;R0=400
2433  006302  000277                SCC                     ;SET CC=0111
2434  006304  000250                CLN
2435  006306  005750                TST     @-(R0)          ;TRY TST W/ MODE 5
```

```
2436  006310  102402                BVS     SNM5A           ;CHECK CC=1000
2437  006312  101401                BLOS    SNM5A
2438  006314  100401                BMI     SNM5B
2439  006316                    SNM5A:
2440  006316  104000                EMT                     ;CC'S NOT SET PROPERLY
2441  006320  005200            SNM5B: INC    R0              ;R0=377
2442  006322  105100                COMB    R0              ;R0=0
2443  006324  001401                BEQ     TS111
2444  006326  104000                EMT                     ;MODE 5 DID NOT DEC R0 CORRECTLY
2445
2446                    ;*****************************************************************************
2447                    ;
2448                    ;       THIS TEST VERIFIES MODE 6 SOP NON-MODIFYING INSTRUCTIONS.
2449                    ;R0 IS SET TO 377 AND A MODE 6 TST INSTRUCTION IS EXECUTED
2450                    ;USING R0 AND AN OFFSET OF -377.  THE CC'S ARE CHECKED AS WELL
2451                    ;AS R0 TO INSURE IT WAS NOT ALTERED.
2452                    ;
2453                    ;*****************************************************************************
2454                    ;TEST 111       TEST MODE 6 W/ SOP NON-MODIFYING INSTS
2455                    ;*****************************************************************************
2456  006330            TS111:
2457  006330  005000            CLR     R0              ;R0=0
2458  006332  0050[illegible]   CLR     (R0)            ;LOC 0=0
2459  006334  0051[illegible]   COM     (R0)            ;LOC 0=-1
2460  006336  105100            COMB    R0              ;R0=377
2461  006340  000277            SCC                     ;SET CC=0111
2462  006342  000250            CLN
2463  006344  005760  177401      TST     -377(R0)        ;TRY TST W/ MODE 6
2464  006350  102402            BVS     SNM6A           ;CHECK CC=1000
2465  006352  101401            BLOS    SNM6A
2466  006354  100401            BMI     SNM6B
2467  006356                    SNM6A:
2468  006356  104000                EMT                     ;CC'S INCORRECT
2469  006360  105100            SNM6B: COMB   R0              ;R0=0
2470  006362  001401                BEQ     TS112
2471  006364  104000                EMT                     ;TST MODE 6 INCORRECTLY CHANGED R0
```

```
2472
2473                    ;*****************************************************************************
2474                    ;
2475                    ;       THIS TEST VERIFIES MODE 7 SOP NON-MODIFYING INSTRUCTIONS.
2476                    ;IT USES A POINTER TO LOC. 0 STORED AT LOC. 400 TO TST LOC. 0.
2477                    ;R0 IS SET TO 377 AND LOC. 0 IS TESTED THRU THE POINTER AT 400 USING
2478                    ;R0 AND AN OFFSET OF 1.
2479                    ;
2480                    ;*****************************************************************************
2481                    ;TEST 112       TEST MODE 7 W/ SOP NON-MODIFYING INSTS.
2482                    ;*****************************************************************************
2483  006366            TS112:
2484  006366  005000           CLR     R0              ;R0=0
2485  006370  005010           CLR     (R0)            ;LOC 0=0
2486  006372  005110           COM     (R0)            ;LOC 0=-1
2487  006374  105100           COMB    R0              ;R0=377
2488  006376  000277           SCC                     ;CC=0111
2489  006400  000250           CLN
2490  006402  005770  000001   TST     @1(R0)          ;TRY TST W/ MODE 7
2491  006406  102402           BVS     SNM7A           ;CHECK CC=1000
2492  006410  101401           BLOS    SNM7A
2493  006412  100401           BMI     SNM7B
2494  006414            SNM7A:
2495  006414  104000           EMT                     ;CC'S NOT CORRECT
2496  006416  105100   SNM7B:  COMB    R0              ;R0=0
2497  006420  001401           BEQ     TS113
2498  006422  104000           EMT                     ;TST MODE 7 INCORRECTLY CHANGED R0
2499
2500                    ;*****************************************************************************
2501                    ;
2502                    ;       THIS TEST VERIFIES MODE 0 DOUBLE OPERAND INSTRUCTIONS.  IT SETS
2503                    ;DATA IN R0 AND R4 AND USES THE ADD INSTRUCTION TO TEST THE DOP
2504                    ;MICROCODE.
2505                    ;
2506                    ;*****************************************************************************
2507                    ;TEST 113       TEST MODE 0 DOUBLE-OPERAND (DOP) INSTS.
2508                    ;*****************************************************************************
2509  006424            TS113:
2510  006424  005000           CLR     R0              ;R0=0
2511  006426  005100           COM     R0              ;R0=-1
2512  006430  005004           CLR     R4              ;R4=0
2513  006432  060004           ADD     R0,R4           ;TRY ADD: R4=-1
2514  006434  005204           INC     R4              ;R4=0
2515  006436  001401           BEQ     TS114
2516  006440  104000           EMT                     ;ADD INST. FAILED W/ MODE 0
2517
2518                    ;*****************************************************************************
2519                    ;
2520                    ;       THIS TEST VERIFIES THE MOVE INSTRUCTION WITH MODE 0 TO MODE 0.
2521                    ;
2522                    ;*****************************************************************************
2523                    ;TEST 114       MOV MODE 0 TO MODE 0
2524                    ;*****************************************************************************
2525  006442            TS114:
2526  006442  005000           CLR     R0              ;R0=0
2527  006444  005004           CLR     R4              ;R4=0
```

```
2528  006446  005100          COM     R0              ;R0=-1
2529  006450  010004          MOV     R0,R4           ;TRY MOVE -1 TO R4
2530  006452  005204          INC     R4              ;INC R4
2531  006454  001401          BEQ     TS115
2532  006456  104000          EMT                     ;MOVE FAILED MODE 0 TO MODE 0
2533
2534                  ;*****************************************************************************
2535                  ;
2536                  ;       THIS TEST VERIFIES THE SUBTRACT INSTRUCTION WITH MODE 0,0.
2537                  ;
2538                  ;*****************************************************************************
2539                  ;TEST 115       TEST SUB MODE 0,0
2540                  ;*****************************************************************************
2541  006460          TS115:
2542  006460  005000          CLR     R0              ;R0=0
2543  006462  005004          CLR     R4              ;R4=0
2544  006464  005204          INC     R4              ;R4=1
2545  006466  160400          SUB     R4,R0           ;TRY SUB 0,0   R0=-1
2546  006470  100003          BPL     SUB0            ;CC=1001
2547  006472  001402          BEQ     SUB0
2548  006474  102401          BVS     SUB0
2549  006476  103401          BCS     SUB0A
2550  006500          SUB0:
2551  006500  104000          EMT                     ;CONDITION CODE FAILED ON SUB
2552  006502  005200  SUB0A:  INC     R0
2553  006504  001401          BEQ     TS116
2554  006506  104000          EMT                     ;DATA RESULT OF SUB FAILED
2555
2556                  ;*****************************************************************************
2557                  ;
2558                  ;       THIS TEST QUICKLY VERIFIES THE REMAINING DOP MODIFYING INSTRUCTIONS
2559                  ;WITH MODE 0,0 TO PROVIDE A BASELINE FOR SUBSEQUENT TESTS.
2560                  ;SINGLE OPERAND INSTRUCTIONS ARE USED TO SET UP DATA IN R0 AND R4
2561                  ;BEFORE EACH OF THE SEVERAL DOP MODIFYING INSTRUCTIONS ARE USED AND
2562                  ;VERIFIED.
2563                  ;
2564                  ;*****************************************************************************
2565                  ;TEST 116       TEST ALL THE DOP INSTRUCTIONS W/ SOURCE MODE 0,0
2566                  ;*****************************************************************************
2567  006510          TS116:
2568  006510  005000          CLR     R0              ;R0=0
2569  006512  010004          MOV     R0,R4           ;TRY MOVE MODE 0,0
2570  006514  001401          BEQ     DOP0A
2571  006516  104000          EMT                     ;Z-BIT NOT SET
2572  006520  005200  DOP0A:  INC     R0              ;R0=1
2573  006522  005100          COM     R0              ;R0=177776
2574  006524  005104          COM     R4              ;R4=177777
2575  006526  040004          BIC     R0,R4           ;TRY BIC: R4=1
2576  006530  005304          DEC     R4              ;R4=0
2577  006532  001401          BEQ     DOP0B
2578  006534  104000          EMT                     ;BIC CLEAR RESULT INCORRECT
2579  006536  050004  DOP0B:  BIS     R0,R4           ;TRY BIS: R4=177777
2580  006540  005204          INC     R4
2581  006542  005204          INC     R4              ;R4=0
2582  006544  001401          BEQ     DOP0C
2583  006546  104000          EMT                     ;RESULT OF BIS INCORRECT
```

```
2584  006550  005000                DOP0C:  CLR     R0              ;R0=0
2585  006552  105100                        COMB    R0              ;R0=377
2586  006554  005004                        CLR     R4              ;R4=0
2587  006556  005104                        COM     R4              ;R4=177777
2588  006560  040004                        BIC     R0,R4           ;R4=177400
2589  006562  060004                        ADD     R0,R4           ;TRY ADD: R4=177777
2590  006564  005204                        INC     R4              ;R4=0
2591  006566  001401                        BEQ     DOP0D
2592  006570  104000                        EMT                     ;RESULT OF ADD INCORRECT
2593  006572  160004                DOP0D:  SUB     R0,R4           ;177401=R4
2594  006574  105404                        NEGB    R4              ;R4=177777
2595  006576  005204                        INC     R4              ;RD=0
2596  006600  001401                        BEQ     TS117
2597  006602  104000                        EMT                     ;RESULT OF SUB INCORRECT
2598
2599                                ;*****************************************************************************
2600                                ;
2601                                ;       THIS TEST VERIFIES MODE 0,X DOUBLE OPERAND INSTRUCTIONS.  IT SETS
2602                                ;DATA IN R0 AND LOCATION 0 AND OPERATES UPON IT USING DOP INSTRUCTIONS.
2603                                ;
2604                                ;*****************************************************************************
2605                                ;TEST 117       TEST MODE 0,X DOUBLE-OPERAND INSTRUCTIONS
2606                                ;*****************************************************************************
2607  006604                        TS117:
2608  006604  005000                        CLR     R0              ;R0=0
2609  006606  005010                        CLR     (R0)            ;LOC. 0=0
2610  006610  105110                        COMB    (R0)            ;LOC. 0=377
2611  006612  005220                        INC     (R0)+           ;LOC. 0=400   R0=2
2612  006614  005400                        NEG     R0              ;R0=-2
2613  006616  060037  000000                ADD     R0,@#0          ;TRY ADD 0,3;  LOC. 0=376
2614  006622  100403                        BMI     DOP03A          ;CC=0001?
2615  006624  001402                        BEQ     DOP03A
2616  006626  102401                        BVS     DOP03A
2617  006630  103401                        BCS     DOP03B
2618  006632                        DOP03A:
2619  006632  104000                        EMT                     ;CC'S NOT SET CORRECTLY
2620  006634  105137  000000        DOP03B: COMB    @#0             ;LOC. 0=1
2621  006640  005337  000000                DEC     @#0             ;LOC. 0=0
2622  006644  001401                        BEQ     TS120
2623  006646  104000                        EMT                     ;DATA RESULT INCORRECT
2624                                ;*****************************************************************************
2625                                ;
2626                                ;       THIS TEST VERIFIES MODE 0,0 DOP NON-MODIFYING INSTRUCTIONS.
2627                                ;R0 AND R4 ARE PRESET TO 0 AND 1 RESPECTIVELY.  COMPARE INSTRUCTIONS ARE
2628                                ;THEN EXECUTED AND CHECKED.  FIRST R4 IS COMPARED TO R0 THEN R0 TO R4.
2629                                ;
2630                                ;*****************************************************************************
2631                                ;TEST 120       TEST DOP NON-MODIFYING INST. W/ SOURCE MODE 0,0
2632                                ;*****************************************************************************
2633  006650                        TS120:
2634  006650  005000                        CLR     R0              ;R0=0
2635  006652  005004                        CLR     R4              ;R4=0
2636  006654  005204                        INC     R4              ;R4=1
2637  006656  020400                        CMP     R4,R0           ;TRY COMPARE R4 TO R0
2638  006660  003001                        BGT     DNM1
2639  006662  104000                        EMT                     ;CC'S NOT CORRECT FOR CMP
```

```
2640  006664  020004                DNM1:   CMP     R0,R4           ;TRY COMPARE R0 TO R4
2641  006666  002401                        BLT     DNM2
2642  006670  104000                        EMT                     ;CC'S NOT CORRECT FOR CMP
2643  006672  005200                DNM2:   INC     R0              ;R0=1
2644  006674  020400                        CMP     R4,R0           ;TRY COMPARE R4=1 TO R0=1
2645  006676  001401                        BEQ     DNM3
2646  006700  104000                        EMT                     ;CC'S NOT CORRECT (Z=1) FOR CMP
2647  006702  005000                DNM3:   CLR     R0              ;R0=0
2648  006704  005100                        COM     R0              ;R0=177777
2649  006706  005004                        CLR     R4              ;R4=0
2650  006710  030004                        BIT     R0,R4           ;TRY BIT R0 TO R4
2651  006712  001401                        BEQ     DNM4
2652  006714  104000                        EMT                     ;CC'S NOT CORRECT FOR BIT
2653  006716  005304                DNM4:   DEC     R4              ;R4=177777
2654  006720  030004                        BIT     R0,R4           ;TRY BIT AGAIN
2655  006722  100401                        BMI     TS121
2656  006724  104000                        EMT                     ;CC'S NOT CORRECT FOR BIT
2657                                ;*******************************************************************************
2658                                ;
2659                                ;       THIS TEST VERIFIES MODE 0,X DOUBLE OPERAND NON-MODIFYING INSTRUCTIONS.
2660                                ;IT SETS DATA IN R0 AND LOCATION 0 AND COMPARES THEM USING DOPNM INSTRUCTIONS.
2661                                ;
2662                                ;*******************************************************************************
2663                                ;TEST 121       TEST MODE 0,X DOUBLE-OPERAND NON-MODIFYING INSTS.
2664                                ;*******************************************************************************
2665  006726                        TS121:
2666  006726  005000                        CLR     R0              ;R0=0
2667  006730  005010                        CLR     (R0)            ;LOC. 0=0
2668  006732  005110                        COM     (R0)            ;LOC. 0=177777
2669  006734  005200                        INC     R0              ;R0=1
2670  006736  020037  000000                CMP     R0,@#0          ;TRY CMP MODE 0,3
2671  006742  100403                        BMI     DNM03A          ;CC=0001
2672  006744  001402                        BEQ     DNM03A
2673  006746  102401                        BVS     DNM03A
2674  006750  103401                        BCS     DNM03B
2675  006752                        DNM03A:
2676  006752  104000                        EMT                     ;CC'S NOT SET CORRECTLY
2677  006754  005300                DNM03B: DEC     R0
2678  006756  001002                        BNE     DNM03C
2679  006760  005210                        INC     (R0)
2680  006762  001401                        BEQ     TS122
2681  006764                        DNM03C:
2682  006764  104000                        EMT                     ;DATA INCORRECTLY MODIFIED BY CMP
2683                                ;*******************************************************************************
2684                                ;
2685                                ;       THIS TEST VERIFIES MODE 1 DOP INSTRUCTIONS.  R0 IS SET TO -1
2686                                ;AND LOC 0 TO 1.  R4 IS THEN CLEARED AND USED TO POINT TO LOC 0.
2687                                ;IN THE ADD MODE 1 INSTRUCTION, LOC 0 IS ADDED TO R0 AND THE
2688                                ;RESULTS VERIFIED.
2689                                ;
2690                                ;*******************************************************************************
2691                                ;TEST 122       TEST MODE 1 W/ DOP INST.
2692                                ;*******************************************************************************
2693  006766                        TS122:
2694  006766  005000                        CLR     R0              ;R0=0
2695  006770  005100                        COM     R0              ;R0=177777
```

```
2696  006772  005004          CLR     R4              ;R4=0
2697  006774  005014          CLR     (R4)            ;LOC 0=0
2698  006776  005214          INC     (R4)            ;LOC 0=1
2699  007000  061400          ADD     (R4),R0         ;TRY ADD SOURCE MODE 1
2700  007002  001401          BEQ     TS123
2701  007004  104000          EMT                     ;RESULT OF ADD INCORRECT
2702
2703                  ;*****************************************************************************
2704                  ;
2705                  ;       THIS TEST VERIFIES MODE 1 DOP BYTE INSTRUCTIONS WHICH ADDRESS
2706                  ;EVEN BYTES.  LOC. 0 IS SET TO -1 AND R4 IS CLEARED.  THEN R4 IS
2707                  ;SET TO -1 USING A BISB THRU R0 WITH MODE 1.
2708                  ;
2709                  ;*****************************************************************************
2710                  ;TEST 123       TEST MODE 1 - EVEN BYTE W/ DOP INSTS.
2711                  ;*****************************************************************************
2712  007006          TS123:
2713  007006  005000          CLR     R0              ;R0=0
2714  007010  005010          CLR     (R0)            ;LOC. 0=0
2715  007012  005110          COM     (R0)            ;LOC. 0=177777
2716  007014  005004          CLR     R4              ;R4=0
2717  007016  151004          BISB    (R0),R4         ;TRY MODE 1- EVEN BYTE W/ DOP
2718  007020  105104          COMB    R4              ;R4=0
2719  007022  001401          BEQ     TS124
2720  007024  104000          EMT                     ;RESULT OF BISB IS INCORRECT
2721
2722                  ;*****************************************************************************
2723                  ;
2724                  ;       THIS TEST VERIFIES MODE 1 DOP NON-MODIFYING INSTRUCTIONS
2725                  ;WHICH ADDRESS EVEN BYTES.  LOC. 0 IS SET TO -1 AND R0 IS CLEARED
2726                  ;AND USED AS THE ADDRESSING REGISTER.  R4 IS SET TO 377 AND A
2727                  ;MODE 1,0 CMPB INSTRUCTION IS USED THE RESULTS VERIFIED.
2728                  ;
2729                  ;*****************************************************************************
2730                  ;TEST 124       TEST MODE 1 - EVEN BYTE W/ DOP NON-MODIFYING INST.
2731                  ;*****************************************************************************
2732  007026          TS124:
2733  007026  005000          CLR     R0              ;R0=0
2734  007030  005010          CLR     (R0)            ;LOC 0=0
2735  007032  005110          COM     (R0)            ;LOC 0=177777
2736  007034  005004          CLR     R4              ;R4=0
2737  007036  105104          COMB    R4              ;R4=377
2738  007040  121004          CMPB    (R0),R4         ;TRY MODE 1 - EVEN BYTE W/ DOP NON-MODIFYING
2739  007042  001401          BEQ     TS125
2740  007044  104000          EMT                     ;RESULT OF CMPB INCORRECT
2741
2742                  ;*****************************************************************************
2743                  ;
2744                  ;       THIS TEST VERIFIES MODE 1,0 MOVB INSTRUCTIONS
2745                  ;WHICH ADDRESS EVEN BYTES.  LOC. 0 IS SET TO 177400, R0 IS CLEARED AND
2746                  ;R4 IS SET TO -1.  MOVB ARE USED TO MOVE BYTE 0 TO R4.  THIS
2747                  ;VERIFIES THAT THE PROPER BYTE WAS SELECTED AND THAT THE SIGN-X-TEND
2748                  ;FUNCTION WITH MODE 0.
2749                  ;       THEN LOC. 0 IS COMPLEMENTED AND THE SAME PROCEDURE EXERCISES
2750                  ;THE LOGIC FOR COMPLEMENTARY DATA.
2751                  ;       THIS TEST EXERCISES UNIQUE MICROCODE.
```

```
                                ;
                                ;*******************************************************************************
                                ;TEST 125       TEST MOV INSTRUCTION MODE 1,0 EVEN BYTE
                                ;*******************************************************************************
                                ;
007046                          TS125:
007046  005000                          CLR     R0              ;R0=0
007050  005010                          CLR     (R0)            ;LOC 0=0
007052  105110                          COMB    (R0)            ;LOC 0=177400
007054  005110                          COM     (R0)
007056  005004                          CLR     R4              ;R4=0
007060  005104                          COM     R4              ;R4=177777
007062  111004                          MOVB    (R0),R4         ;R4=0
007064  005704                          TST     R4              ;CHECK SIGN OF WORD
007066  001401                          BEQ     DOP1
007070  104000                          EMT                     ;MOVB SHOULD SIGN X-TEND
007072  005110                  DOP1:   COM     (R0)            ;LOC 0=177777
007074  111004                          MOVB    (R0),R4         ;DO MOVB W/ EVEN BYTE
007076  100401                          BMI     TS126
007100  104000                          EMT                     ;MOVB SHOULD SIGN X-TEND
                                ;
                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 1 DOP INSTRUCTIONS WHICH REFERENCE
                                ;ODD BYTES.  LOC. 0 IS SET TO 177400.  R0 IS SET TO 0 AND R4 IS
                                ;SET TO 1.  THE BISB INSTRUCTION USES THE DATA IN BYTE 1 TO SET BYTE 0.
                                ;THE RESULT IS CHECKED BY INCREMENTING THE WORD (LOC. 0) TO ZERO.
                                ;
                                ;*******************************************************************************
                                ;TEST 126       TEST MODE 1-ODD BYTE W/ DOP INSTS.
                                ;*******************************************************************************
007102                          TS126:
007102  005000                          CLR     R0              ;R0=0
007104  005010                          CLR     (R0)            ;LOC. 0=0
007106  005004                          CLR     R4              ;R4=0
007110  005204                          INC     R4              ;R4=1
007112  105114                          COMB    (R4)            ;LOC. 0=177400
007114  151410                          BISB    (R4),(R0)       ;TRY TO BIS LOW ORDER BITS W/ MODE 1
007116  005210                          INC     (R0)            ;CHECK RESULT
007120  001401                          BEQ     TS127
007122  104000                          EMT                     ;RESULT OF BISB INCORRECT
                                ;
                                ;*******************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 2 DOP INSTRUCTIONS.  LOC. 0 IS SET TO -1.
                                ;R0 IS CLEARED AND USED AS THE MODE 2 ADDRESSING REGISTER TO MOVE LOC. 0
                                ;TO R7.  THE DATA RESULTS ARE VERIFIED AND THE INCREMENTING OF THE REGISTER
                                ;IS CHECKED.
                                ;
                                ;*******************************************************************************
                                ;TEST 127       TEST MODE 2 W/ DOP INSTS.
                                ;*******************************************************************************
007124                          TS127:
007124  005000                          CLR     R0              ;R0=0
007126  005010                          CLR     (R0)            ;LOC. 0=0
007130  005110                          COM     (R0)            ;LOC. 0=177777
007132  012004                          MOV     (R0)+,R4        ;TRY MOVE MODE 2,0
```

```
2808  007134  005204                INC     R4              ;CHECK R4
2809  007136  001401                BEQ     DOP2
2810  007140  104000                EMT                     ;RESULT OF MOV INST INCORRECT
2811  007142  005300        DOP2:   DEC     R0              ;TEST R0 AFTER MODE 2
2812  007144  005300                DEC     R0
2813  007146  001401                BEQ     TS130
2814  007150  104000                EMT                     ;REGISTER NOT INCREMENTED IN MODE 2
2815
2816                  ;*****************************************************************************
2817                  ;
2818                  ;       THIS TEST VERIFIES MODE 2 DOP BYTE INSTRUCTIONS WHICH ADDRESS
2819                  ;EVEN BYTES.  LOC. 0 IS SET TO -1.  R0 IS CLEARED AND USED AS THE
2820                  ;ADDRESSING REGISTER IN A TEST WHICH TRIES TO CLEAR BYTE 1 USING
2821                  ;BYTE 0 DATA AND A BICB.  UNIQUE IN THIS TEST IS USE OF THE
2822                  ;SAME ADDRESSING REGISTER FOR BOTH SOURCE AND DESTINATION.  THE SOURCE AND
2823                  ;DESTINATION IS CHECKED TO INSURE PROPER FUNCTIONING.
2824                  ;
2825                  ;*****************************************************************************
2826                  ;TEST 130       TEST MODE 2 - EVEN BYTE W/ DOP INST.
2827                  ;*****************************************************************************
2828  007152          TS130:
2829  007152  005000                CLR     R0              ;R0=0
2830  007154  010010                MOV     R0,(R0)         ;LOC. 0=0
2831  007156  005110                COM     (R0)            ;LOC. 0=177777
2832  007160  142010                BICB    (R0)+,(R0)      ;TRY TO CLEAR BYTE 1 FROM BYTE 0 W/ BICB
2833  007162  105737  000001        TSTB    @#1             ;CHECK RESULT
2834  007166  001401                BEQ     DOPB2A
2835  007170  104000                EMT                     ;BICB DESTINATION INCORRECT
2836  007172  105137  000000  DOPB2A: COMB    @#0             ;CHECK BICB SOURCE
2837  007176  001401                BEQ     TS131
2838  007200  104000                EMT                     ;BICB SOURCE INCORRECTLY CHANGED
2839                  ;*****************************************************************************
2840                  ;
2841                  ;       THIS TEST VERIFIES MODE 2 DOP BYTE INSTRUCTIONS WHICH REFERENCE
2842                  ;ODD BYTES.  R0 IS SET TO 1, LOC. 0 IS SET TO 177400, AND R4 IS CLEARED.
2843                  ;A MODE 2 MOVB USES R0 TO MOVE BYTE 1 TO R4.  AN INCREMENT
2844                  ;IS USED TO CHECK THAT THE PROPER BYTE WAS MOVED AND SIGN X-TENDED.
2845                  ;
2846                  ;*****************************************************************************
2847                  ;TEST 131       TEST MODE 2 - ODD BYTE W/ DOP INST.
2848                  ;*****************************************************************************
2849  007202          TS131:
2850  007202  005000                CLR     R0              ;R0=0
2851  007204  005004                CLR     R4              ;R4=0
2852  007206  005010                CLR     (R0)            ;LOC. 0=0
2853  007210  005110                COM     (R0)            ;LOC. 0=177777
2854  007212  105120                COMB    (R0)+           ;LOC 0=177400; R0=1
2855  007214  112004                MOVB    (R0)+,R4        ;TRY DOP MODE 2 W/ ODD BYTE
2856  007216  005204                INC     R4              ;CHECK RESULT OF MOVB
2857  007220  001401                BEQ     DOPB2B
2858  007222  104000                EMT                     ;RESULT OF MOVB INCORRECT
2859  007224  005740  DOPB2B: TST     -(R0)           ;BUMP R0 DOWN BY 2
2860  007226  005700                TST     R0              ;CHECK R0
2861  007230  001401                BEQ     TS132
2862  007232  104000                EMT                     ;MODE 2 BYTE DID NOT INCREMENT REG. CORRECTLY
2863                  ;*****************************************************************************
```

```
                                ;
                                ;       THIS TEST VERIFIES MODE 3 DOUBLE-OPERAND INSTRUCTIONS.
                                ;LOC. 0 IS LOADED WITH ALTERNATING ZEROES AND ONES; AND R0 IS LOADED
                                ;WITH ALTERNATING ONES AND ZEROES.  A MODE 3 BIS IS USED TO SET R0
                                ;TO -1 BY USING LOC. 0 AS THE SOURCE TO BIS THE ZEROES IN R0.  THE
                                ;RESULT IS TESTED BY INCREMENTING R0 AND CHECKING FOR ZERO.
                                ;
                                ;*****************************************************************************
                                ;TEST 132       TEST MODE 3 W/ DOP INSTS.
                                ;*****************************************************************************
007234                          TS132:
007234  012737  052525  000000          MOV     #052525,@#0     ;MOVE 52525 TO LOC. 0
007242  012700  125252                  MOV     #125252,R0      ;SET ALT. ONE AND ZERO IN R0
007246  053700  000000                  BIS     @#0,R0          ;TRY TO SET ALL OTHER BITS W/ MODE 3
007252  005200                          INC     R0              ;TEST RESULT
007254  001401                          BEQ     TS133
007256  104000                          EMT                     ;BIS W/ MODE 3 INCORRECT RESULT
                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 3 DOUBLE OPERAND BYTE INSTRUCTIONS WHICH
                                ;ADDRESS EVEN BYTES.  BYTE 0 IS SET TO ALTERNATING 1'S AND 0'S; BYTE 1,
                                ;ALTERNATING 0'S AND 1'S.  R0 IS CLEARED AND A BISB IS USED TO
                                ;SET THE LOW BYTE OF R0 TO 252.
                                ;
                                ;*****************************************************************************
                                ;TEST 133       TEST MODE 3 - EVEN BYTE W/ DOP INSTS.
                                ;*****************************************************************************
007260                          TS133:
007260  012737  052652  000000          MOV     #52652,@#0      ;NOVE 1'S AND 0' PATTERN TO LOC. 0
007266  005000                          CLR     R0              ;R0=0
007270  153700  000000                  BISB    @#0,R0          ;TRY R0=252 W/ MODE 3 - EVEN BYTE
007274  022700  000252                  CMP     #252,R0         ;BISB W/ EVEN BYTE SUCCESSFUL?
007300  001401                          BEQ     TS134
007302  104000                          EMT                     ;BISB W/ MODE 3 - EVEN BYTE FAILED
                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 3 DOUBLE OPERAND BYTE INSTRUCTIONS
                                ;WHICH ADDRESS ODD BYTES.  THE SAME PROCEDURE USED IN PREVIOUS
                                ;TEST IS USED HERE.  THIS TIME BYTE 1 IS USED AS THE SOURCE BYTE.
                                ;THE EXPECTED RESULT IS:  R0 = 125.
                                ;
                                ;*****************************************************************************
                                ;TEST 134       TEST MODE 3 - ODD BYTE W/ DOP INSTS.
                                ;*****************************************************************************
007304                          TS134:
007304  012737  052652  000000          MOV     #52652,@#0      ;MOVE 1'S AND 0'S PATTERN TO LOC 0
007312  005000                          CLR     R0              ;R0=0
007314  153700  000001                  BISB    @#1,R0          ;TRY R0=152 W/ MODE 3 - ODD BYTE
007320  022700  000125                  CMP     #125,R0         ;R0=125?
007324  001401                          BEQ     TS135
007326  104000                          EMT                     ;BISB W/ MODE 3 - ODD BYTE FAILED
                                ;*****************************************************************************
                                ;TEST 135       TEST DEST. MODE 0-BYTE W/ DOP NON-MODIFYING MST
                                ;*****************************************************************************
007330                          TS135:
```

```
2920  007330  005000                  CLR     R0              ;R0=0
2921  007332  105100                  COMB    R0              ;R0=377
2922  007334  000263                  +SEC!SEV                ;SET C AND V BITS
2923  007336  132700  000200          BITB    #200,R0         ;TRY DOPNM DEST. MODE 0-BYTE
2924  007342  001403                  BEQ     DNMB0A          ;BR TO ERROR IF Z BIT SET
2925  007344  102402                  BVS     DNMB0A          ;BR TO ERROR IF V BIT SET
2926  007346  103001                  BCC     DNMB0A          ;BR TO ERROR IF C BIT CLEAR.
2927  007350  100401                  BMI     DNMB0B
2928  007352                  DNMB0A:
2929  007352  104000                  EMT                     ;CC'S INCORRECT
2930  007354  105100          DNMB0B: COMB    R0              ;CHECK DESTINATION DATA
2931  007356  001401                  BEQ     TS136
2932  007360  104000                  EMT                     ;DEST. DATA MODIFIED
2933
2934                          ;*****************************************************************************
2935                          ;TEST 136       TEST DEST. MODE 1 W/ DOP NON-MODIFYING INST
2936                          ;*****************************************************************************
2937  007362                  TS136:
2938  007362  005000                  CLR     R0              ;R0=0
2939  007364  005010                  CLR     (R0)            ;LOC. 0=0
2940  007366  000241                  CLC                     ;CLEAR C BIT
2941  007370  032710  177777          BIT     #177777,(R0)    ;TRY DOPNM DEST. MODE 1
2942  007374  100403                  BMI     DNM1A           ;BR TO ERROR IF N BIT SET
2943  007376  102402                  BVS     DNM1A           ;BR TO ERROR IF V BIT SET
2944  007400  103401                  BCS     DNM1A           ;BR TO ERROR IF C BIT SET
2945  007402  001401                  BEQ     DNM1B
2946  007404                  DNM1A:
2947  007404  104000                  EMT                     ;COND. CODES INCORRECT
2948  007406  005710          DNM1B:  TST     (R0)            ;CHECK TEST DATA
2949  007410  001401                  BEQ     TS137
2950  007412  104000                  EMT                     ;DESTINATION DATA MODIFIED
2951
2952                          ;*****************************************************************************
2953                          ;TEST 137       TEST DEST, MODE 2 W/ DOP NON-MODIFYING INST.
2954                          ;*****************************************************************************
2955  007414                  TS137:
2956  007414  005000                  CLR     R0              ;R0=0
2957  007416  005010                  CLR     (R0)            ;LOC. 0=0
2958  007420  052710  125252          BIS     #125252,(R0)    ;LOC. 0=125252
2959  007424  032720  077777          BIT     #77777,(R0)+    ;TRY DOPNM INST W/ MODE 2
2960  007430  102402                  BVS     DNM2A           ;BR TO ERROR IF V BIT SET
2961  007432  001401                  BEQ     DNM2A           ;BR TO ERROR IF Z-BIT SET
2962  007434  100001                  BPL     DNM2B
2963  007436                  DNM2A:
2964  007436  104000                  EMT                     ;COND. CODES INCORRECT
2965  007440  005300          DNM2B:  DEC     R0              ;DECREMENT R0 TO CHECK IT.
2966  007442  005300                  DEC     R0
2967  007444  001401                  BEQ     DNM2D
2968  007446                  DNM2C:
2969  007446  104000                  EMT                     ;MODE 2 REGISTER NOT INCREMENTED BY 2
2970  007450  022710  125252  DNM2D:  CMP     #125252,(R0)    ;CHECK DEST. DATA
2971  007454  001401                  BEQ     TS140
2972  007456  104000                  EMT                     ;DEST. DATA MODIFIED
2973
2974                          ;*****************************************************************************
2975                          ;TEST 140       TEST DEST. MODE 2-BYTE, W/DOP NON-MODIFYING INST
```

```
2976
2977  007460
2978  007460  005000                ;*****************************************************************************
2979  007462  005010                TS140:
2980  007464  052710  052652              CLR     R0              ;R0=0
2981  007470  000263                      CLR     (R0)            ;LOC. 0=0
2982  007472  132720  000201              BIS     #52652,(R0)     ;LOC. 0=52652
2983  007476  001403                      +SEC!SEV                ;SET C AND V BITS
2984  007500  103002                      BITB    #201,(R0)+      ;TRY DOPNM INST. W/ MODE 2 EVEN BYTE
2985  007502  102401                      BEQ     DNMB2A          ;BR TO ERROR IF Z-BIT SET
2986  007504  100401                      BCC     DNMB2A          ;BR TO ERROR IF C-BIT CLEAR
2987  007506                              BVS     DNMB2A          ;BR TO ERROR IF V-BIT SET
2988  007506  104000                      BMI     DNMB2B
2989  007510  005300              DNMB2A:
2990  007512  001401                      EMT                     ;COND. CODES INCORRECT
2991  007514  104000              DNMB2B: DEC     R0              ;CHECK DEST. REGISTER.
2992  007516  005200                      BEQ     DNMB2C
2993  007520  132720  000201              EMT                     ;DEST. REGISTER NOT INCREMENTED BY 1
2994  007524  001402              DNMB2C: INC     R0              ;R0=1
2995  007526  102401                      BITB    #201,(R0)+      ;TRY DOPNM INST. W/MODE 2-ODD BYTE
2996  007530  100001                      BEQ     DNMB2D  ;BR TO ERROR IF Z-BIT SET
2997  007532                              BVS     DNMB2D          ;BR TO ERROR IF V-BIT SET
2998  007532  104000                      BPL     DNMB2E
2999  007534  005300              DNMB2D:
3000  007536  005300                      EMT                     ;COND. CODES INCORRECT
3001  007540  001401              DNMB2E: DEC     R0              ;DEC R0 TO CHECK IT.
3002  007542  104000                      DEC     R0
3003  007544  022710  052652              BEQ     DNMB2F
3004  007550  001401                      EMT                     ;DEST. REGISTER NOT INCREMENTED BY 1
3005  007552  104000              DNMB2F: CMP     #52652,(R0)     ;CHECK DEST. DATA IS UNMODIFIED
3006                                      BEQ     TS141
3007                                      EMT                     ;DEST. DATA WAS MODIFIED.
3008
3009                              ;*****************************************************************************
3010                              ;TEST 141       TEST DEST. MODE 3-BYTES W/DOP NON-MODIFYING INST.
3011  007554                      ;*****************************************************************************
3012  007554  005000              TS141:
3013  007556  005010                      CLR     R0              ;R0=0
3014  007560  052710  125125              CLR     (R0)            ;LOC. 0=0
3015  007564  105100                      BIS     #125125,(R0)    ;LOC. 0=125125
3016  007566  005200                      COMB    R0              ;R0=377
3017  007570  005010                      INC     R0              ;R0=400
3018  007572  000263                      CLR     (R0)            ;LOC. 400=0
3019  007574  132730  000201              +SEC!SEV                ;C-BIT=V-BIT=1
3020  007600  001403                      BITB    #201,@(R0)+     ;TRY DOPNM W/MODE 3-EVEN BYTE
3021  007602  102402                      BEQ     DNMB3A          ;BR TO ERROR IF Z BIT SET
3022  007604  103001                      BVS     DNMB3A          ;BR TO ERROR IF V BIT SET
3023  007606  100001                      BCC     DNMB3A          ;BR TO ERROR IF C BIT CLEAR
3024  007610                              BPL     DNMB3B
3025  007610  104000              DNMB3A:
3026  007612  022700  000402              EMT                     ;COND. CODES INCORRECT
3027  007616  001401              DNMB3B: CMP     #402,R0         ;CHECK DEST. REGISTER INC. BY 2 AND INC BY 2 AGAIN
3028  007620  104000                      BEQ     DNMB3C
3029  007622  005200                      EMT                     ;DEST. REGISTER NOT INCREMENTED BY 2
3030  007624  005200              DNMB3C: INC     R0              ;R0=404
3031  007626  132730  000201              INC     R0
                                      BITB    #201,@(R0)+     ;TRY DOPNM DEST MODE 3-BYTE(ODD)
```

```
3032  007632  001402                        BEQ     DNMB3D          ;BR TO ERROR IF Z BIT SET
3033  007634  102401                        BVS     DNMB3D          ;BR TO ERROR IF V BIT SET
3034  007636  100401                        BMI     DNMB3E
3035  007640                        DNMB3D:
3036  007640  104000                        EMT                     ;COND. CODES INCORRECT
3037  007642  005004                DNMB3E: CLR     R4              ;R4=0
3038  007644  022714  125125                CMP     #125125,(R4)    ;CHECK DEST. DATA
3039  007650  001401                        BEQ     TS142
3040  007652  104000                        EMT                     ;DEST. DATA MODIFIED
3041
3042                                ;*****************************************************************************
3043                                ;TEST 142       TEST DEST. MODE 4 W/DOP NON-MODIFYING INST.
3044                                ;*****************************************************************************
3045  007654                        TS142:
3046  007654  005000                        CLR     R0              ;R0=0
3047  007656  005010                        CLR     (R0)            ;LOC. 0=0
3048  007660  052710  125252                BIS     #125252,(R0)    ;LOC. 0=125125
3049  007664  052700  000002                BIS     #2,R0           ;R0=2
3050  007670  000277                        SCC                     ;SET ALL COND. CODE BITS
3051  007672  032740  020000                BIT     #20000,-(R0)    ;TRY DOPNM W/ MODE 4
3052  007676  100403                        BMI     DNM4A           ;BR TO ERROR IF N-BIT SET
3053  007700  102402                        BVS     DNM4A           ;BR TO ERROR IF V-BIT SET
3054  007702  103001                        BCC     DNM4A           ;BR TO ERROR IF C-BIT CHAR
3055  007704  001001                        BNE     DNM4B
3056  007706                        DNM4A:
3057  007706  104000                        EMT                     ;COND. CODES INCORRECT
3058  007710  005700                DNM4B:  TST     R0              ;CHECK DEST. REGISTER
3059  007712  001401                        BEQ     DNM4C
3060  007714  104000                        EMT                     ;DEST. REGISTER NOT DECREMENTED BY 2
3061  007716  022737  125252  000000 DNM4C:  CMP     #125252,@#0     ;CHECK DEST. DATA
3062  007724  001401                        BEQ     TS143
3063  007726  104000                        EMT                     ;DEST. DATA MODIFIED
3064
3065                                ;*****************************************************************************
3066                                ;TEST 143       TEST DEST. MODE 4-BYTE W/ DOP NON-MODIFYING INST.
3067                                ;*****************************************************************************
3068  007730                        TS143:
3069  007730  005000                        CLR     R0              ;R0=0
3070  007732  005010                        CLR     (R0)            ;LOC. 0=0
3071  007734  052710  052652                BIS     #52652,(R0)     ;LOC. 0=52652
3072  007740  052700  000002                BIS     #2,R0           ;R0=2
3073  007744  000257                        CCC                     ;COND. CODES=0
3074  007746  132740  000201                BITB    #201,-(R0)      ;TRY DOPNM INST W/MODE 4 ODD BYTE
3075  007752  102403                        BVS     DNMB4A          ;BR TO ERROR IF V BIT SET
3076  007754  001402                        BEQ     DNMB4A  ;BR TO ERROR IF Z BIT SET
3077  007756  103401                        BCS     DNMB4A          ;BR TO ERROR IF C BIT SET
3078  007760  001001                        BNE     DNMB4B
3079  007762                        DNMB4A:
3080  007762  104000                        EMT                     ;COND. CODES INCORRECT
3081  007764  022700  000001        DNMB4B: CMP     #1,R0           ;CHECK DEST. REGISTER
3082  007770  001401                        BEQ     DNMB4C
3083  007772  104000                        EMT                     ;DEST REG. NOT DECREMENTED BY 1
3084  007774  132740  000201        DNMB4C: BITB    #201,-(R0)      ;TRY DOPNM INST. W/MODE 4 EVEN BYTE
3085  010000  001401                        BEQ     DNMB4D          ;BR TO ERROR IF Z-BIT SET
3086  010002  100401                        BMI     DNMB4E
3087  010004                        DNMB4D:
```

```
3088  010004  104000                        EMT                     ;COND. CODES INCORRECT
3089  010006  005700                DNMB4E: TST     R0              ;CHECK DEST. REGISTER
3090  010010  001401                        BEQ     DNMB4F
3091  010012  104000                        EMT                     ;DEST. REG. NOT DECREMENTED BY 1
3092  010014  022710  052652        DNMB4F: CMP     #52652,(R0)     ;CHECK DESTINATION DATA
3093  010020  001401                        BEQ     TS144
3094  010022  104000                        EMT                     ;DEST. DATA MODIFIED
3095
3096                                ;*****************************************************************************
3097                                ;TEST 144       TEST DEST MODE 5 W/DOP NON-MODIFYING INST.
3098                                ;*****************************************************************************
3099  010024                        TS144:
3100  010024  005000                        CLR     R0              ;R0=0
3101  010026  005010                        CLR     (R0)            ;LOC 0=0
3102  010030  052710  100000                BIS     #100000,(R0)    ;LOC. 0=100000
3103  010034  052700  000402                BIS     #402,R0         ;R0=2
3104  010040  000277                        SCC                     ;SET ALL COND. CODE BITS
3105  010042  032750  100000                BIT     #100000,@-(R0)  ;TRY DOPNM W/MODE 5
3106  010046  102403                        BVS     DNM5A           ;BR TO ERROR IF V-BIT SET
3107  010050  103002                        BCC     DNM5A           ;BR TO ERROR IF C-BIT CLEAR
3108  010052  001401                        BEQ     DNM5A           ;BR TO ERROR IF Z-BIT SET
3109  010054  100401                        BMI     DNM5B
3110  010056                        DNM5A:
3111  010056  104000                        EMT                     ;COND. CODES INCORRECT
3112  010060  022700  000400        DNM5B:  CMP #400,R0             ;CHECK DEST. REGISTER
3113  010064  001401                        BEQ     DNM5C
3114  010066  104000                        EMT                     ;DEST. REGISTER NOT DECREMENTED BY 2
3115  010070  022737  100000  000000 DNM5C: CMP     #100000,@#0     ;CHECK DESTINATION DATA
3116  010076  001401                        BEQ     TS145
3117  010100  104000                        EMT                     ;DEST. DATA INCORRECTLY MODIFIED
3118
3119                                ;*****************************************************************************
3120                                ;TEST 145       TEST DEST. MODE 6 W/DOP NON-MODIFYING INST.
3121                                ;*****************************************************************************
3122  010102                        TS145:
3123  010102  005000                        CLR     R0              ;R0=0
3124  010104  005010                        CLR     (R0)            ;LOC> 0=0
3125  010106  052710  000001                BIS     #1,(R0)         ;LOC. 0=1
3126  010112  005100                        COM     R0              ;R0=-1 C-BIT=1
3127  010114  032760  000001  000001        BIT     #1,1(R0)        ;TRY DOPNM W/MODE 6
3128  010122  001403                        BEQ     DNM6A           ;BR TO ERROR IF Z-BIT SET
3129  010124  102402                        BVS     DNM6A           ;BR TO ERROR IF V-BIT SET
3130  010126  103001                        BCC     DNM6A           ;BR TO ERROR IF C-BIT CLEAR
3131  010130  100001                        BPL     DNM6B
3132  010132                        DNM6A:
3133  010132  104000                        EMT                     ;COND CODES INCORRECT
3134  010134  022700  177777        DNM6B:  CMP     #-1,R0          ;CHECK DEST. REGISTER
3135  010140  001401                        BEQ     DNM6C
3136  010142  104000                        EMT                     ;DEST. REGISTER MODIFIED
3137  010144  022737  000001  000000 DNM6C: CMP     #1,@#0          ;CHECK DEST. DATA
3138  010152  001401                        BEQ     TS146
3139  010154  104000                        EMT                     ;DEST. DATA MODIFIED
3140
3141                                ;*****************************************************************************
3142                                ;TEST 146       TEST DEST MODE 7 W/DOP NON-MODIFYING INST.
3143                                ;*****************************************************************************
```

```
3144  010156
3145  010156  005000                  TS146:
3146  010160  005010                          CLR     R0              ;R0=0
3147  010162  052710  125125                  CLR     (R0)            ;LOC. 0=0 C-BIT=0
3148  010166  052700  000001                  BIS     #125125,(R0)    ;LOC. 0=125125
3149  010172  132770  000125  000403         BIS     #1,R0           ;R0=1
3150  010200  102403                          BITB    #125,@403(R0)   ;TRY DOPNM W/MODE 7
3151  010202  100402                          BVS     DNM7A           ;BR TO ERROR IF V-BIT SET
3152  010204  103401                          BMI     DNM7A           ;BR TO ERROR IF N-BIT SET
3153  010206  001401                          BCS     DNM7A           ;BR TO ERROR IF C-BIT SET
3154  010210                                   BEQ     DNM7B
3155  010210  104000                  DNM7A:
                                               EMT                     ;COND. CODES INCORRECT
3156  010212  022700  000001          DNM7B:  CMP     #1,R0           ;CHECK DEST. REGISTER
3157  010216  001401                          BEQ     DNM7C
3158  010220  104000                          EMT                     ;DESTINATION REGISTER MODIFIED
3159  010222  022737  125125  000000  DNM7C:  CMP     #125125,@#0     ;CHECK DEST. DATA
3160  010230  001401                          BEQ     TS147
3161  010232  104000                          EMT                     ;DEST. DATA INCORRECT
3162
3163                                   ;*****************************************************************************
3164                                   ;
3165                                   ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 1 INSTRUCTION.
3166                                   ;DATA IS SET IN R0 USING SOP INSTRUCTIONS AND THEN MOVED TO LOC. 0
3167                                   ;USING MOV SRC MODE 0, DEST. MODE 1.
3168                                   ;
3169                                   ;*****************************************************************************
3170                                   ;TEST 147       TEST MOV DESTINATION MODE 1
3171                                   ;*****************************************************************************
3172  010234                           TS147:
3173  010234  005000                          CLR     R0              ;R0=0
3174  010236  005010                          CLR     (R0)            ;LOC. 0=0
3175  010240  005100                          COM     R0              ;R0=-1
3176  010242  005004                          CLR     R4              ;R4 POINTS TO LOC. 0
3177  010244  010014                          MOV     R0,(R4)         ;TRY MOVE MODE 0,1
3178  010246  102402                          BVS     MDM1A           ;BR TO ERROR IF V SET
3179  010250  001401                          BEQ     MDM1A           ;BR TO ERROR IF Z SET
3180  010252  100401                          BMI     MDM1B
3181  010254                           MDM1A:
3182  010254  104000                          EMT                     ;CONDITION CODE NOT CORRECT
3183  010256  005704                  MDM1B:  TST     R4
3184  010260  001401                          BEQ     TS150
3185  010262  104000                          EMT                     ;DESTINATION REGISTER INCORRECTLY ALTERED
3186
3187                                   ;*****************************************************************************
3188                                   ;
3189                                   ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 2 INSTRUCTION.
3190                                   ;       DATA IS SET IN R0 USING SOP INSTRUCTIONS AND THEN MOVED
3191                                   ;       TO LOCATION 0 USING MOV SRC MODE 0, DEST. MODE 1.
3192                                   ;
3193                                   ;*****************************************************************************
3194                                   ;TEST 150       TEST MOV DESTINATION MODE 2
3195                                   ;*****************************************************************************
3196  010264                           TS150:
3197  010264  005000                          CLR     R0              ;R0=0
3198  010266  005001                          CLR     R1              ;R1=0
3199  010270  005010                          CLR     (R0)            ;LOC.0=0
```

```
3200  010272  005110                        COM     (R0)            ;LOC. 0= 1
3201  010274  010120                        MOV     R1,(R0)+        ;TRY MOVE MODE 0,2
3202  010276  100402                        BMI     MDM2A           ;BR TO ERROR IF N SET
3203  010300  102401                        BVS     MDM2A           ;BR TO ERROR IF V SET
3204  010302  001401                        BEQ     MDM2B
3205  010304                        MDM2A:
3206  010304  104000                        EMT                     ;CC'S INCORRECT
3207  010306  005300                MDM2B:  DEC     R0
3208  010310  005300                        DEC     R0
3209  010312  001401                        BEQ     MDM2D
3210  010314                        MDM2C:
3211  010314  104000                        EMT                     ;DESTINATION REGISTER NOT INCREMENTED PROPERLY
3212  010316  005737  000000        MDM2D:  TST     @#0
3213  010322  001401                        BEQ     TS151
3214  010324  104000                        EMT                     ;DESTINATION DATA INCORRECT
3215
3216                                ;*******************************************************************************
3217                                ;
3218                                ;       THIS TEST VERIFIES DESTINATION MODE 2 W/MOVB INSTS. TWO DIFFERENT MOVB
3219                                ;INSTRUCTIONS ARE USED TO MOVE A TEST PATTERN FIRST TO BYTE 0 THEN TO BYTE 1.
3220                                ;
3221                                ;*******************************************************************************
3222                                ;TEST 151       TEST MOV-BYTE DESTINATION MODE 2
3223                                ;*******************************************************************************
3224  010326                        TS151:
3225  010326  005000                        CLR     R0              ;R0=0
3226  010330  005010                        CLR     (R0)            ;LOC. 0=0
3227  010332  112720  000125               MOVB    #125,(R0)+      ;TRY DESTINATION MODE 2 W/EVEN BYTE
3228  010336  102402                        BVS     MBDM2A          ;BR TO ERROR IF V SET
3229  010340  001401                        BEQ     MBDM2A          ;BR TO ERROR IF Z SET
3230  010342  100001                        BPL     MBDM2B
3231  010344                        MBDM2A:
3232  010344  104000                        EMT                     ;CC'S INCORRECT
3233  010346  022700  000001        MBDM2B: CMP     #1,R0
3234  010352  001401                        BEQ     MBDM2C
3235  010354  104000                        EMT                     ;REGISTER NOT INCREMENTED BY ONE
3236  010356  112720  000252        MBDM2C: MOVB    #252,(R0)+      ;TRY DESTINATION MODE 2 W/ODD BYTE
3237  010362  102402                        BVS     MBDM2D
3238  010364  001401                        BEQ     MBDM2D
3239  010366  100401                        BMI     MBDM2E
3240  010370                        MBDM2D:
3241  010370  104000                        EMT                     ;CC'S NOT SET CORRECT
3242  010372  022700  000002        MBDM2E: CMP     #2,R0
3243  010376  001401                        BEQ     MBDM2F
3244  010400  104000                        EMT                     ;REGISTER NOT INCREMENTED BY ONE
3245  010402  022737  125125  000000 MBDM2F: CMP     #125125,@#0     ;CHECK DATA
3246  010410  001401                        BEQ     TS152
3247  010412  104000                        EMT                     ;DESTINATION DATA INCORRECT
3248
3249                                ;*******************************************************************************
3250                                ;
3251                                ;       THIS TEST VERIFIES MOV DESTINATION MODE 3. R0 IS USED TO PICK UP
3252                                ;AN ADDRESS AT LOC. 400. LOC 400 POINTS TO LOC. 0 THE EFFECTIVE DEST. ADDR.. ALSO, MOVB
3253                                ;INST. ARE USED W/ EVEN AND ODD BYTES TO CHECK MOV BYTES INST AND MODE 37 DESTINATIONS.
3254                                ;*******************************************************************************
3255                                ;TEST 152       TEST MOV(B) DESTINATION MODE 3
```

```
3256                                    ;*****************************************************************************
3257  010414                            TS152:
3258  010414  012700  000400                   MOV     #400,R0         ;R0=400
3259  010420  005010                           CLR     (R0)            ;LOC. 400 POINTS TO LOC. 0
3260  010422  005037  000000                   CLR     @#0             ;LOC. 0=0
3261  010426  012730  125252                   MOV     #125252,@(R0)+  ;TRY MOV DESTINATION MODE 2
3262  010432  102402                           BVS     MDM3A           ;BR TO ERROR IF V SET
3263  010434  001401                           BEQ     MDM3A           ;BR TO ERROR IF Z SET
3264  010436  100401                           BMI     MDM3B
3265  010440                            MDM3A:
3266  010440  104000                           EMT                     ;CC'S INCORRECT
3267  010442  022700  000402           MDM3B:  CMP     #402,R0         ;CHECK DEST. MODE REGISTER
3268  010446  001401                           BEQ     MDM3C
3269  010450  104000                           EMT                     ;REGISTER NOT INCREMENTED BY 2
3270  010452  022737  125252  000000   MDM3C:  CMP     #125252,@#0     ;CHECK DESTINATION DATA
3271  010460  001401                           BEQ     MDM3D
3272  010462  104000                           EMT                     ;DESTINATION DATA INCORRECT
3273  010464  112737  000125  000000   MDM3D:  MOVB    #125,@#0        ;TRY MOVB DESTINATION MODE 2 EVEN BYTE
3274  010472  022737  125125  000000           CMP     #125125,@#0     ;CHECK DATA
3275  010500  001401                           BEQ     MDM3E
3276  010502  104000                           EMT                     ;DESTINATION DATA INCORRECT
3277  010504  112737  000525  000001   MDM3E:  MOVB    #525,@#1        ;TRY MOVB DESTINATION MODE 2 ODD BYTE
3278  010512  022737  052525  000000           CMP     #52525,@#0      ;CHECK DATA
3279  010520  001401                           BEQ     TS153
3280  010522  104000                           EMT                     ;
3281                                    ;*****************************************************************************
3282                                    ;*****************************************************************************
3283                                    ;
3284                                    ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 4 INSTRUCTION.
3285                                    ;SOP INSTRUCTIONS ON R0 ARE USED TO CLEAR TARGET LOCATION 0.
3286                                    ;R4 IS USED AS THE MODE 4 ADDRESSING REGISTER, AND
3287                                    ;CONDITIONAL BRANCHES ARE USED TO VERIFY THE DATA.
3288                                    ;
3289                                    ;*****************************************************************************
3290                                    ;TEST 153       TEST MOV DESTINATION MODE 4
3291                                    ;*****************************************************************************
3292  010524                            TS153:
3293  010524  005000                           CLR     R0              ;R0=0
3294  010526  005010                           CLR     (R0)            ;LOC. 0=0
3295  010530  012704  000002                   MOV     #2,R4           ;R4=2
3296  010534  012744  012345                   MOV     #12345,-(R4)    ;TRY MOV DEST. MODE 4
3297  010540  102402                           BVS     MDM4A           ;BR TO ERROR IF V-BIT SET
3298  010542  001401                           BEQ     MDM4A           ;BR TO ERROR IF Z-BIT SET
3299  010544  100001                           BPL     MDM4B
3300  010546                            MDM4A:
3301  010546  104000                           EMT                     ;CC'S NOT CORRECT
3302  010550  005704                   MDM4B:  TST     R4              ;CHECK DECREMENTING OF MODE 4 REG.
3303  010552  001401                           BEQ     MDM4C
3304  010554  104000                           EMT                     ;DESTINATION MODE REGISTER NOT DECREMENTED BY 2
3305  010556  022710  012345           MDM4C:  CMP     #12345,(R0)     ;CHECK DESTINATION DATA
3306  010562  001401                           BEQ     TS154
3307  010564  104000                           EMT                     ;DESTINATION DATA INCORRECT
3308                                    ;*****************************************************************************
3309                                    ;*****************************************************************************
3310                                    ;
3311                                    ;       THIS TEST VERIFIES THE MOVB DESTINATION MODE 4 INSTRUCTION
```

```
                        ;       ON BOTH ODD AND EVEN BYTES. SOP INSTRUCTIONS ON R4 ARE
                        ;       USED TO CLEAR TARGET LOCATION 0. R0 IS USED AS THE MODE 4
                        ;       ADDRESSING REGISTER, AND CMP AND CONDITIONAL BRANCH
                        ;       INSTRUCTIONS ARE USED TO VERIFY THE DATA.
                        ;
                        ;*****************************************************************************************
                        ;TEST 154       TEST MOVB DESTINATION MODE 4
                        ;*****************************************************************************************
010566                  TS154:
010566  005004                  CLR     R4              ;R4=0
010570  005014                  CLR     (R4)            ;LOC. 0=0
010572  012700  000002          MOV     #2,R0           ;R0 = 2
010576  112740  125125          MOVB    #125125,-(R0)   ;TRY MOVB DEST. MODE 4-ODD BYTE
010602  020027  000001          CMP     R0,#1           ;CHECK THAT DEST. REG. WAS DECREMENTED
010606  001401                  BEQ     MBDM4A
010610  104000                  EMT                     ;DESTINATION REG. NOT DECREMENTED BY 1
010612  021427  052400  MBDM4A: CMP     (R4),#52400     ;CHECK DEST. DATA
010616  001401                  BEQ     MBDM4B
010620  104000                  EMT                     ;DEST. DATA NOT CORRECT
010622  112740  125125  MBDM4B: MOVB    #125125,-(R0)   ;TRY MOVB DEST. MODE 4--EVEN BYTE
010626  102402                  BVS     MBDM4C          ;BR. TO ERROR IF V-BIT SET
010630  001401                  BEQ     MBDM4C          ;BR TO ERROR IF Z-BIT SET
010632  100001                  BPL     MBDM4D
010634  MBDM4C:
010634  104000                  EMT                     ;COND. CODES INCORRECT
010636  005700  MBDM4D: TST     R0                      ;CHECK MODE 4 DEST. REGISTER
010640  001401                  BEQ     MBDM4E
010642  104000                  EMT                     ;DESTINATION REG NOT DECREMENTED BY 1
010644  021427  052525  MBDM4E: CMP     (R4),#52525     ;CHECK DEST. DAT^A
010650  001401                  BEQ     TS155
010652  104000                  EMT                     ;DESTINATION DATA INCORRECT

                        ;*****************************************************************************************
                        ;
                        ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 5 AND THE MOVB
                        ;DESTINATION MODE 5 - EVEN BYTE INSTRUCTIONS. R4 IS A
                        ;POINTER TO TARGET LOCATION 0 AND R0 IS SETUP TO
                        ;POINT TO LOCATION 376 FOR THE MOV, AND LOCATION 404 FOR
                        ;THE MOVB INSTRUCTIONS. CMP INSTRUCTIONS ARE USED TO VERIFY
                        ;PROPER ADDRESSING AND DATA.
                        ;
                        ;*****************************************************************************************
                        ;TEST 155       TEST MOV DESTINATION MODE 5
                        ;*****************************************************************************************
010654                  TS155:
010654  005004                  CLR     R4              ;R4=0
010656  005014                  CLR     (R4)            ;LOC. 0 = 0
010660  012700  000400          MOV     #400,R0         ;R0=400
010664  012750  004321          MOV     #4321,@-(R0)    ;TRY MOV DEST. MODE 5
010670  102402                  BVS     MDM5A           ;BR TO ERROR IF V-BIT SET
010672  001401                  BEQ     MDM5A           ;BR TO ERROR IF Z-BIT SET
010674  100001                  BPL     MDM5B
010676  MDM5A:
010676  104000                  EMT                     ;COND. CODES INCORRECT
010700  022700  000376  MDM5B: CMP     #376,R0         ;CHECK MODE 5 REG. WAS DECREMENTED
010704  001401                  BEQ     MDM5C
```

```
3368  010706  104000                        EMT                     ;MODE 5 REGISTER NOT DECREMENTED BY 2
3369  010710  022714  004321        MDM5C:  CMP     #4321,(R4)      ;CHECK DEST. DATA
3370  010714  001401                        BEQ     MDM5D
3371  010716  104000                        EMT                     ;DEST. DATA INCORRECT
3372  010720  012700  000406        MDM5D:  MOV     #406,R0         ;R0=406
3373  010724  112750  000377                MOVB    #377,@-(R0)     ;TRY MOV DEST. MODE 5 --EVEN BYTE
3374  010730  022700  000404                CMP     #404,R0         ;CHECK MODE 5 REG.
3375  010734  001401                        BEQ     MDM5E
3376  010736  104000                        EMT                     ;MODE 5 REGISTER NOT DECREMENTED BY 2
3377  010740  022714  177721        MDM5E:  CMP     #177721,(R4)    ;CHECK DEST. DATA
3378  010744  001401                        BEQ     TS156
3379  010746  104000                        EMT                     ;DEST. DATA INCORRECT
3380
3381                                ;*****************************************************************************
3382                                ;
3383                                ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 6 AND MOVB - EVEN BYTE
3384                                ;DESTINATION MODE 6 INDTRUCTIONS. R0 IS USED TO SETUP TARGET LOC.0
3385                                ;FOR BOTH TESTS. PATTERNS OF ONES AND ZEROES ARE MOVED INTO LOC.0
3386                                ;BY MODE 6 INSTRUCTIONS, AND CMP INSTRUCTIONS ARE USED TO VERIFY
3387                                ;PROPER ADDRESSING AND DATA.
3388                                ;
3389                                ;*****************************************************************************
3390                                ;TEST 156       TEST MOV DESTINATION MODE 6
3391                                ;*****************************************************************************
3392  010750                        TS156:
3393  010750  005000                        CLR     R0              ;R0=0
3394  010752  005010                        CLR     (R0)            ;LOC. 0=0
3395  010754  005200                        INC     R0              ;R0=1
3396  010756  012760  052525  177777        MOV     #052525,-1(R0)  ;TRY MOV DEST. MODE 6
3397  010764  102402                        BVS     MDM6A           ;BR TO ERROR IF V-BIT SET
3398  010766  001401                        BEQ     MDM6A           ;BR TO ERROR IF Z-BIT SET
3399  010770  100001                        BPL     MDM6B
3400  010772                        MDM6A:
3401  010772  104000                        EMT                     ;COND. CODES INCORRECT
3402  010774  022700  000001        MDM6B:  CMP     #1,R0           ;CHECK DEST. REGISTER UNALTERED
3403  011000  001401                        BEQ     MDM6C
3404  011002  104000                        EMT                     ;DEST. REGISTER INCORRECTLY ALTERED
3405  011004  022737  052525  000000  MDM6C:  CMP     #52525,@#0      ;CHECK DEST. DATA
3406  011012  001401                        BEQ     MDM6D
3407  011014  104000                        EMT                     ;DEST. DATA INCORRECT
3408  011016  012700  000002        MDM6D:  MOV     #2,R0           ;R0=2
3409  011022  112760  000377  177777        MOVB    #377,-1(R0)     ;TRY MOVB DEST. MODE 6
3410  011030  022700  000002                CMP     #2,R0           ;CHECK DEST. REGISTER UNALTERED
3411  011034  001401                        BEQ     MDM6E
3412  011036  104000                        EMT                     ;DEST. REGISTER INCORRECTLY ALTERED
3413  011040  022737  177525  000000  MDM6E:  CMP     #177525,@#0     ;CHECK DEST. DATA
3414  011046  001401                        BEQ     TS157
3415  011050  104000                        EMT                     ;DEST. DATA INCORRECT
3416
3417                                ;*****************************************************************************
3418                                ;
3419                                ;       THIS TEST VERIFIES THE MOV DESTINATION MODE 7 AND MOVB - ODD BYTE
3420                                ;DESTINATION MODE 7 INSTRUCTIONS. R4 POINTS TO TARGET LOC.0 AND R0
3421                                ;IS USED AS THE MODE 7 ADDRESSING REGISTER. CMP INSTRUCTIONS ARE
3422                                ;USED TO VERIFY PROPER ADDRESSING AND DATA.
3423                                ;
```

```
                                        ;*****************************************************************************
                                        ;TEST 157       TEST MOV DESTINATION MODE 7
                                        ;*****************************************************************************
                                        TS157:
011052                                          CLR     R4              ;R4=0
011052  005004                                  CLR     (R4)            ;LOC.0=0
011054  005014                                  MOV     #403,R0         ;R0=403
011056  012700  000403                          MOV     #70707,@-1(R0)  ;TRY MOV W/DEST MODE 7
011062  012770  070707  177777                  BVS     MDM7A           ;BR. TO ERROR IF V-BIT SET
011070  102402                                  BEQ     MDM7A           ;BR TO ERROR IF Z-BIT SET
011072  001401                                  BPL     MDM7B
011074  100001                          MDM7A:
011076                                          EMT                     ;COND. CODES INCORRECT
011076  104000
011100  022700  000403                  MDM7B:  CMP     #403,R0         ;CHECK DEST. REGISTER
011104  001401                                  BEQ     MDM7C
011106  104000                                  EMT                     ;DEST. REGISTER INCORRECTLY ALTERED
011110  022737  070707  000000          MDM7C:  CMP     #70707,@#0      ;CHECK DEST. DATA
011116  001401                                  BEQ     MDM7D
011120  104000                                  EMT                     ;DEST. DATA INCORRECT
011122  112770  107070  000001          MDM7D:  MOVB    #107070,@1(R0)  ;TRY MOVB W/DEST MODE 7--ODD BYTE
011130  022700  000403                          CMP     #403,R0         ;CHECK MODE 7 DEST. REG.
011134  001401                                  BEQ     MDM7E
011136  104000                                  EMT                     ;DEST. DATA INCORRECT
011140  022737  034307  000000          MDM7E:  CMP     #34307,@#0      ;CHECK DEST. DATA
011146  001401                                  BEQ     TS160
011150  104000                                  EMT                     ;DESTINATION DATA INCORRECT

                                        ;*****************************************************************************
                                        ;
                                        ;       THIS TEST VERIFIES MODE 4 DOUBLE OPERAND INSTRUCTIONS.
                                        ;THE TEST USES MODE 4 ADDRESSING WITH REGISTER 0 TO MOVE THRU A
                                        ;TABLE OF OPERANDS.  THE TABLE OF OPERANDS AND THE WORK LOCATION IS
                                        ;STORED FOLLOWING THE TEST CODE.  A SERIES OF 5 DOP INSTRUCTIONS UTILIZES
                                        ;THE DATA IN THE TABLE TO CYCLE THE WORK LOCATION THRU A SET OF
                                        ;VALUE.  THE DATA HAS BEEN CHOSEN TO INSURE THAT NO SINGLE ERROR WILL
                                        ;GO UNDETECTED.  WORD AND BYTE INSTRUCTION ACCESSING BOTH EVEN AND
                                        ;ODD ADDRESSES ARE USED IN THE TEST.  THE LISTING SHOWS THE
                                        ;EXPECTED INTERMEDIATE RESULT AS EACH INSTRUCTION IS EXECUTED.
                                        ;
                                        ;*****************************************************************************
                                        ;TEST 160       TEST MODE 4 W/ DOP INSTS.
                                        ;*****************************************************************************
                                        TS160:
011152                                          MOV     #TBL1,R0        ;INITIALIZE R0
011152  012700  011216                          MOV     -(R0),@#TBL1    ;TBL1=125252
011156  014037  011216                          ADD     -(R0),@#TBL1    ;TBL1=000377
011162  064037  011216                          BICB    -(R0),@#TBL1    ;TBL1=000252
011166  144037  011216                          BISB    -(R0),@#TBL1+1  ;TBL1=125252
011172  154037  011217                          CMP     -(R0),@#TBL1    ;CHECK RESULT
011176  024037  011216                          BEQ     TS161
011202  001406                          DOP4:
011204                                          EMT                     ;RESULT OF MODE 4 INSTS. INCORRECT
011204  104000

011206  125252                                  125252
011210  052652                                  52652
011212  053125                                  53125
```

```
3480  011214  125252                    125252
3481  011216  000000            TBL1:   0
3482
3483                            ;*****************************************************************************
3484                            ;
3485                            ;       THIS TEST VERIFIES MODE 5 DOUBLE OPERAND INSTRUCTIONS.
3486                            ;THE TEST USES AN ADDRESS TABLE STORED FOLLOWING THE TEST CODE.
3487                            ;THIS TABLE IS SIMPLY A TABLE OF ADDRESS POINTERS WHICH ADDRESS
3488                            ;THE DATA TABLE USED IN THE PREVIOUS TEST.  THE TEST IS IDENTICAL TO
3489                            ;THE PREVIOUS TEST EXCEPT THE DATA IS REFERENCED USING THIS ADDRESS
3490                            ;TABLE AND MODE 5 ADDRESSING.  (SEE PREVIOUS TEST).
3491                            ;
3492                            ;*****************************************************************************
3493                            ;TEST 161       TEST MODE 5 W/ DOP INSTS.
3494                            ;*****************************************************************************
3495  011220                    TS161:
3496  011220  012700  011266           MOV     #TBL2+2,R0      ;INITIALIZE R0
3497  011224  015037  011216           MOV     @-(R0),@#TBL1   ;TBL1=125252
3498  011230  065037  011216           ADD     @-(R0),@#TBL1   ;TBL1=000377
3499  011234  145037  011216           BICB    @-(R0),@#TBL1   ;TBL1=000252
3500  011240  155037  011217           BISB    @-(R0),@#TBL1+1 ;TBL1=125252
3501  011244  025037  011216           CMP     @-(R0),@#TBL1   ;CHECK RESULT
3502  011250  001406                   BEQ     TS162
3503  011252                    DOP5:
3504  011252  104000                   EMT                     ;RESULT OF MODE 5 INSTS. INCORRECT
3505  011254  011206                   TBL1-10
3506  011256  011210                   TBL1-6
3507  011260  011211                   TBL1-5
3508  011262  011212                   TBL1-4
3509  011264  011214            TBL2:   TBL1-2
3510                            ;*****************************************************************************
3511                            ;
3512                            ;       THIS TEST VERIFIES MODE 6 DOUBLE OPERAND INSTRUCTIONS.
3513                            ;IT USES THE SAME DATA AS THAT USED IN THE MODE 4 TESTS.
3514                            ;THIS TIME THE DATA IS ACCESSED USING MODE 6.  R0 IS SET
3515                            ;TO POINT TO THE MIDDLE OF THE TABLE.  THE TABLE IS ACCESSED FROM
3516                            ;BOTTOM TO TOP BY VARYING THE OFFSET IN THE MODE 6 INSTRUCTIONS.
3517                            ;THE DATA RESULTS ARE IDENTICAL TO THOSE EXPECTED IN THE MODE 4
3518                            ;TESTS.
3519                            ;
3520                            ;*****************************************************************************
3521                            ;TEST 162       TEST MODE 6 W/ DOP INSTS.
3522                            ;*****************************************************************************
3523  011266                    TS162:
3524  011266  012700  011212           MOV     #TBL1-4,R0      ;INITIALIZE R0
3525  011272  016037  000002  011216   MOV     2(R0),@#TBL1    ;TBL1=125252
3526  011300  066037  000000  011216   ADD     0(R0),@#TBL1    ;TBL1=000377
3527  011306  146037  177777  011216   BICB    -1(R0),@#TBL1   ;TBL1=000252
3528  011314  156037  177776  011217   BISB    -2(R0),@#TBL1+1 ;TBL1=125252
3529  011322  026037  177774  011216   CMP     -4(R0),@#TBL1   ;CHECK RESULT
3530  011330  001401                   BEQ     TS163
3531  011332  104000                   EMT                     ;RESULT OF MODE 6 INSTS. INCORRECT
3532                            ;*****************************************************************************
3533                            ;
3534                            ;       THIS TEST VERIFIES MODE 7 DOUBLE OPERAND INSTRUCTIONS.
3535                            ;THIS TEST USES THE SAME ADDRESS TABLE AND DATA TABLE USED BY
```

```
3536                                    ;THE MODE 5 TESTS.  THIS TIME THE DATA IS ACCESSED USING MODE 7.
3537                                    ;R0 IS SET TO POINT TO THE MIDDLE OF THE ADDRESS TABLE IN THE MODE 5
3538                                    ;TEST.  THE TABLE IS ACCESSED FROM BOTTOM TO TOP BY VARYING THE OFFSET
3539                                    ;IN THE MODE 7 INSTRUCTIONS.  THE DATA RESULTS ARE IDENTICAL TO
3540                                    ;THOSE EXPECTED IN THE MODE 5 TESTS.
3541                                    ;
3542                                    ;*****************************************************************************
3543                                    ;TEST 163       TEST MODE 7 W/ DOP INSTS.
3544                                    ;*****************************************************************************
3545  011334                            TS163:
3546  011334  012700  011260                   MOV     #TBL2-4,R0      ;INITIALIZE R0
3547  011340  017037  000004  011216           MOV     @4(R0),@#TBL1   ;TBL1=125252
3548  011346  067037  000002  011216           ADD     @2(R0),@#TBL1   ;TBL1=000377
3549  011354  147037  000000  011216           BICB    @0(R0),@#TBL1   ;TBL1=000252
3550  011362  157037  177776  011217           BISB    @-2(R0),@#TBL1+1                ;TBL1=125252
3551  011370  027037  177774  011216           CMP     @-4(R0),@#TBL1          ;CHECK RESULT
3552  011376  001401                           BEQ     TS164
3553  011400  104000                           EMT                             ;RESULT OF MODE 7 INSTS INCORRECT
3554                                    ;*****************************************************************************
3555                                    ;
3556                                    ;       THIS TEST VERIFIES THE ROTATE MODE 0 INSTRUCTIONS.
3557                                    ;R0 IS LOADED WITH A DATA PATTERN, THE C-BIT IS LOADED, AND
3558                                    ;AN ROL INSTRUCTION IS EXECUTED WITH MODE 0.  THE OPERATION IS CHECKED
3559                                    ;BY TESTING THE RESULTING DATA AND THE STATE OF THE C AND V BITS.
3560                                    ;NEXT, THE SAME PROCEDURE IS EXECUTED TO TEST MODE 0 BYTE INSTRUCTIONS.
3561                                    ;
3562                                    ;*****************************************************************************
3563                                    ;TEST 164       TEST ROTATE INSTRUCTIONS OF MODE 0
3564                                    ;*****************************************************************************
3565  011402                            TS164:
3566  011402  012700  125252                   MOV     #125252,R0      ;INITIALIZE DATA
3567  011406  000261                           SEC                     ;SET C-BIT
3568  011410  006100                           ROL     R0              ;TRY ROL W/ MODE 0
3569  011412  102004                           BVC     ROT0A           ;CC=0011
3570  011414  103003                           BCC     ROT0A
3571  011416  022700  052525                   CMP     #052525,R0      ;CHECK DATA
3572  011422  001401                           BEQ     ROT0B
3573  011424                            ROT0A:
3574  011424  104000                           EMT                     ;ROL MODE 0 FAILED
3575  011426  012700  125252           ROT0B:  MOV     #125252,R0      ;INITIALIZE DATA
3576  011432  000261                           SEC                     ;SET C-BIT
3577  011434  106100                           ROLB    R0              ;TRY ROL W/ MODE 0 EVEN BYTE
3578  011436  102004                           BVC     ROT0C           ;CC=0011
3579  011440  103003                           BCC     ROT0C
3580  011442  022700  125125                   CMP     #125125,R0      ;CHECK DATA
3581  011446  001401                           BEQ     TS165
3582  011450                            ROT0C:
3583  011450  104000                           EMT                     ;ROLB MODE 0 FAILED
3584
3585                                    ;*****************************************************************************
3586                                    ;
3587                                    ;       THIS TEST VERIFIES THE ROTATE MODE 1 INSTRUCTIONS.
3588                                    ;THE DATA TO BE ROTATED IS IN LOC 0.  R0 IS USED AS THE
3589                                    ;ADDRESSING REGISTER.  THE C-BIT IS LOADED AND AN ROL IS EXECUTED.
3590                                    ;THE RESULTS ARE CHECKED BY COMPARING THE DATA RESULTS AND TESTING
3591                                    ;THE C AND V BITS.  THIS PROCEDURE IS THEN REPEATED TWICE MORE
```

```
3592                                    ;TO TEST THE BYTE ROTATES.  FIRST ON BYTE 0, THEN ON BYTE 1.
3593                                    ;
3594                                    ;*****************************************************************************
3595                                    ;TEST 165       TEST ROTATE INSTRUCTIONS W/ MODE 1
3596                                    ;*****************************************************************************
3597  011452                            TS165:
3598  011452  005000                           CLR     R0              ;POINT TO LOC. 0
3599  011454  012710  052525                   MOV     #52525,(R0)     ;INITIALIZE DATA
3600  011460  000241                           CLC                     ;CLEAR C-BIT
3601  011462  006110                           ROL     (R0)            ;TRY ROL W/ MODE 1
3602  011464  102005                           BVC     ROT1A           ;CC=1010
3603  011466  103404                           BCS     ROT1A
3604  011470  023727  000000  125252           CMP     @#0,#125252     ;CHECK RESULT
3605  011476  001401                           BEQ     ROT1B
3606  011500                            ROT1A:
3607  011500  104000                           EMT                     ;ROL MODE 1 FAILED
3608  011502  000261                    ROT1B: SEC
3609  011504  012710  125252                   MOV     #125252,(R0)    ;INITIALIZE DATA
3610  011510  106110                           ROLB    (R0)            ;TRY ROLB W/ MODE 1 EVEN BYTE
3611  011512  102005                           BVC     ROT1C           ;CC=1011
3612  011514  103004                           BCC     ROT1C
3613  011516  022737  125125  000000           CMP     #125125,@#0     ;TEST RESULT
3614  011524  001401                           BEQ     ROT1D
3615  011526                            ROT1C:
3616  011526  104000                           EMT                     ;ROLB W/ MODE 1 EVEN BYTE FAILED
3617  011530  012710  125252           ROT1D:  MOV     #125252,(R0)
3618  011534  005000                           CLR     R0              ;POINT TO ODD BYTE
3619  011536  005200                           INC     R0
3620  011540  000261                           SEC                     ;SET C-BIT
3621  011542  106110                           ROLB    (R0)            ;TRY ROLB W/ MODE 1 ODD BYTE
3622  011544  102005                           BVC     ROT1E           ;CC=0011
3623  011546  103004                           BCC     ROT1E
3624  011550  022737  052652  000000           CMP     #052652,@#0     ;CHECK DATA
3625  011556  001401                           BEQ     TS166
3626  011560                            ROT1E:
3627  011560  104000                           EMT                     ;ROLB W/ MODE 1 ODD BYTE FAILED
3628
3629                                    ;*****************************************************************************
3630                                    ;
3631                                    ;       THIS TEST VERIFIES MODE 2 ROTATE INSTRUCTIONS.
3632                                    ;THE SAME PROCEDURE AS IN THE OTHER ROTATE TESTS ARE USED.  R0
3633                                    ;IS USED AS THE ADDRESSING REGISTER AND IS CHECKED FOR PROPER
3634                                    ;INCREMENTING.  BYTE INSTRUCTIONS ARE ALSO CHECKED.
3635                                    ;
3636                                    ;*****************************************************************************
3637                                    ;TEST 166       TEST ROTATE INSTRUCTIONS W/ MODE 2
3638                                    ;*****************************************************************************
3639  011562                            TS166:
3640  011562  005000                           CLR     R0              ;POINT TO LOC 0
3641  011564  012710  173737                   MOV     #173737,(R0)    ;INITIALIZE DATA
3642  011570  000241                           CLC                     ;CLEAR C-BIT
3643  011572  006120                           ROL     (R0)+           ;TRY ROL W/ MODE 2
3644  011574  103007                           BCC     ROT2A           ;CHECK C-BIT
3645  011576  022737  167676  000000           CMP     #167676,@#0     ;CHECK DATA
3646  011604  001003                           BNE     ROT2A           ;BRANCH IF RESULT INCORRECT
3647  011606  005300                           DEC     R0              ;TEST R0
```

```
3648  011610  005300                          DEC     R0
3649  011612  001401                          BEQ     ROT2B
3650  011614                            ROT2A:
3651  011614  104000                          EMT                     ;ROL W/ MODE 2 FAILED
3652  011616  005000                    ROT2B: CLR     R0              ;POINT TO LOC 0
3653  011620  012710  004040                  MOV     #4040,(R0)      ;INITIALIZE DATA
3654  011624  000241                          CLC                     ;CLEAR C-BIT
3655  011626  106120                          ROLB    (R0)+           ;TRY ROLB W/ MODE 2 EVEN BYTE
3656  011630  103406                          BCS     ROT2C           ;CHECK C-BIT
3657  011632  022737  004100  000000          CMP     #4100,@#0       ;CHECK DATA
3658  011640  001002                          BNE     ROT2C           ;BRANCH IF DATA INCORRECT
3659  011642  005300                          DEC     R0              ;CHECK R0
3660  011644  001401                          BEQ     ROT2D
3661  011646                            ROT2C:
3662  011646  104000                          EMT                     ;ROLB W/ MODE 2 EVEN BYTE FAILED
3663  011650  005000                    ROT2D: CLR     R0              ;POINT TO LOC 0
3664  011652  012710  004040                  MOV     #4040,(R0)      ;INITIALIZE DATA
3665  011656  005200                          INC     R0              ;POINT TO ODD BYTE OF DATA
3666  011660  000261                          SEC                     ;SET C-BIT
3667  011662  106120                          ROLB    (R0)+           ;TRY ROL W/ MODE 2 ODD BYTE
3668  011664  103407                          BCS     ROT2E           ;CHECK C-BIT
3669  011666  022737  010440  000000          CMP     #10440,@#0      ;CHECK DATA
3670  011674  001003                          BNE     ROT2E           ;BRANCH IF DATA INCORRECT
3671  011676  005300                          DEC     R0              ;CHECK R0
3672  011700  005300                          DEC     R0
3673  011702  001401                          BEQ     TS167
3674  011704                            ROT2E:
3675  011704  104000                          EMT                     ;ROLB W/ MODE 2 ODD BYTE FAILED
3676
3677                                    ;*****************************************************************************
3678                                    ;
3679                                    ;       THIS TEST VERIFIES MODE 3 ROTATE INSTRUCTIONS.
3680                                    ;THIS TEST USES THE SAME PROCEDURES AS IN THE OTHER ROTATE
3681                                    ;TESTS.  THE DATA IS STORED IN LOC. 0 AND IS ADDRESSED USING
3682                                    ;MODE 37.  BYTE ADDRESSING IS ALSO CHECKED FOR EVEN AND ODD BYTES.
3683                                    ;
3684                                    ;*****************************************************************************
3685                                    ;TEST 167       TEST ROTATE INSTRUCTIONS /W MODE 3
3686                                    ;*****************************************************************************
3687  011706                            TS167:
3688  011706  012737  052525  000000          MOV     #52525,@#0      ;INITIALIZE DATA IN LOC 0
3689  011714  000261                          SEC                     ;SET C-BIT
3690  011716  006137  000000                  ROL     @#0             ;TRO ROL W/ MODE 3
3691  011722  103404                          BCS     ROT3A           ;CHECK C-BIT
3692  011724  022737  125253  000000          CMP     #125253,@#0     ;CHECK DATA
3693  011732  001401                          BEQ     ROT3B
3694  011734                            ROT3A:
3695  011734  104000                          EMT                     ;ROL W/ MODE 3 FAILED
3696  011736  012737  125252  000000    ROT3B: MOV     #125252,@#0     ;INITIALIZE DATA
3697  011744  000241                          CLC                     ;CLEAR C-BIT
3698  011746  106137  000000                  ROLB    @#0             ;TRY ROL W/ MODE 3 EVEN BYTE
3699  011752  103004                          BCC     ROT3C           ;CHECK C-BIT
3700  011754  023727  000000  125124    4$:   CMP     @#0,#125124     ;CHECK DATA
3701  011762  001401                          BEQ     ROT3D
3702  011764                            ROT3C:
3703  011764  104000                          EMT                     ;ROL W/ MODE 3 EVEN BYTE FAILED
```

```
3704  011766  012737  125252  000000  ROT3D:  MOV     #125252,@#0     ;INITIALIZE DATA IN LOC. 0
3705  011774  000261                          SEC                     ;SET C-BIT
3706  011776  106137  000001                  ROLB    @#1             ;TRY ROL W/ MODE 3 ODD BYTE
3707  012002  103004                          BCC     ROT3E           ;CHECK C-BIT
3708  012004  022737  052652  000000          CMP     #052652,@#0     ;CHECK DATA
3709  012012  001401                          BEQ     TS170
3710  012014                          ROT3E:
3711  012014  104000                          EMT                     ;ROL W/ MODE 3 ODD BYTE FAILED
3712
3713                                  ;*******************************************************************************
3714                                  ;
3715                                  ;       THIS TEST VERIFIES MODE 4 ROTATE INSTRUCTIONS.  THE DATA IS
3716                                  ;STORED IN LOC. 0.  R0 IS SET TO 2 AND THE CARRY IS SET.  AN ROL MODE 4
3717                                  ;IS USED TO ROTATE LOCATION 0 USING R0.  THE DATA IS CHECKED
3718                                  ;AND THE C AND V BITS ARE TESTED.  THE PROPER DECREMENTING OF
3719                                  ;R0 IS VERIFIED.
3720                                  ;
3721                                  ;*******************************************************************************
3722                                  ;TEST 170       TEST MODE 4 W/ ROTATE INSTRUCTIONS
3723                                  ;*******************************************************************************
3724  012016                          TS170:
3725  012016  012737  070707  000000          MOV     #070707,@#0     ;INITIALIZE DATA IN LOC. 0
3726  012024  012700  000002                  MOV     #2,R0           ;INITIALIZE R0 AS POINTER
3727  012030  000261                          SEC                     ;SET C-BIT
3728  012032  006140                          ROL     -(R0)           ;TRY ROL W/ MODE 4
3729  012034  103406                          BCS     ROT4            ;CHECK C-BIT
3730  012036  022737  161617  000000          CMP     #161617,@#0     ;CHECK DATA
3731  012044  001002                          BNE     ROT4            ;BRANCH IF DATA INCORRECT
3732  012046  005700                          TST     R0              ;CHECK MODE 4 REGISTER
3733  012050  001401                          BEQ     TS171
3734  012052                          ROT4:
3735  012052  104000                          EMT                     ;ROL MODE 4 FAILED
3736
3737                                  ;*******************************************************************************
3738                                  ;
3739                                  ;       THIS TEST VERIFIES MODE 5 ROTATE INSTRUCTIONS.
3740                                  ;THE DATA IS STORED IN A WORK LOCATION (ROTX) AT THE END OF THE
3741                                  ;TEST CODE.  LOC. 0 IS LOADED WITH THE ADDRESS OF THE DATA (ROTX).
3742                                  ;R0 IS SET TO 2.  THE CARRY IS CLEARED AND A MODE 5 ROL
3743                                  ;IS EXECUTED USING R0 AS AN ADDRESSING REGISTER.  THE DATA IS
3744                                  ;CHECKED, THE C AND V BITS TESTED, AND R0 CHECKED FOR PROPER
3745                                  ;DECREMENTING.
3746                                  ;
3747                                  ;*******************************************************************************
3748                                  ;TEST 171       TEST MODE 5 W/ ROTATE INSTRUCTIONS
3749                                  ;*******************************************************************************
3750  012054                          TS171:
3751  012054  012737  012120  000000          MOV     #ROTX,@#0       ;MOVE POINTER TO LOC. 0
3752  012062  012700  000002                  MOV     #2,R0           ;SET MODE 5 REG. TO LOC. 0
3753  012066  012767  107070  000024          MOV     #107070,ROTX    ;INITIALIZE DATA
3754  012074  000241                          CLC                     ;CLEAR C-BIT
3755  012076  006150                          ROL     @-(R0)          ;TRY ROL W/ MODE 5
3756  012100  103006                          BCC     ROT5            ;CHECK C-BIT
3757  012102  022737  016160  012120          CMP     #016160,@#ROTX  ;CHECK DATA
3758  012110  001002                          BNE     ROT5            ;BRANCH IF DATA INCORRECT
3759  012112  005700                          TST     R0              ;CHECK MODE 5 REGISTER
```

```
3760  012114  001402                        BEQ     TS172
3761  012116                        ROT5:
3762  012116  104000                        EMT                     ;ROL MODE 5 FAILED
3763  012120  000000                ROTX:   0
3764
3765                                ;*****************************************************************************
3766                                ;
3767                                ;       THIS TEST VERIFIES MODE 6 ROTATE INSTRUCTIONS.
3768                                ;IT USES THE SAME PROCEDURE AS THE ABOVE TEST EXCEPT THE
3769                                ;ROTATE INSTRUCTION USES MODE 6 ADDRESSING WITH REGISTER 7.
3770                                ;THE DATA IS STILL OPERATED ON IN LOC. ROTX (SEE PREVIOUS TEST).
3771                                ;
3772                                ;*****************************************************************************
3773                                ;TEST 172       TEST MODE 6 W/ ROTATE INSTRUCTIONS
3774                                ;*****************************************************************************
3775  012122                        TS172:
3776  012122  012737  125252  012120        MOV     #125252,@#ROTX  ;INITIALIZE DATA
3777  012130  000261                        SEC                     ;SET C-BIT
3778  012132  006167  177762                ROL     ROTX            ;TRY ROL W/ MODE 6
3779  012136  103004                        BCC     ROT6            ;CHECK C-BIT
3780  012140  022737  052525  012120        CMP     #52525,@#ROTX   ;CHECK DATA
3781  012146  001401                        BEQ     TS173
3782  012150                        ROT6:
3783  012150  104000                        EMT                     ;ROL W/ MODE 6 FAILED
3784
3785                                ;*****************************************************************************
3786                                ;
3787                                ;THIS TEST VERIFIES MODE 7 ROTATE INSTRUCTIONS.
```

```
                                ;THE DATA IS SET IN LOC. ROTX, (SEE PREVIOUS TEST).  THE ROL INSTRUCTION
                                ;ADDRESSES IT INDIRECTLY USING MODE 7 AND INDIRECT ADDRESS LOCATION
                                ;(ROTXAD) FOLLOWING THE TEST CODE.
                                ;
                                ;*****************************************************************************
                                ;TEST 173        TEST MODE 7 W/ ROTATE INSTRUCTIONS
                                ;*****************************************************************************
012152                          TS173:
012152  012737  052525  012120          MOV     #52525,@#ROTX   ;INITIALIZE DATA
012160  012737  012120  012210          MOV     #ROTX,@#ROTXAD  ;INITIALIZE ADDRESS POINTER
012166  000241                          CLC                     ;CLEAR C-BIT
012170  006177  000014                  ROL     @ROTXAD         ;TRY ROL W/ MODE 7
012174  103404                          BCS     ROT7            ;CHECK C-BIT
012176  023727  012120  125252          CMP     @#ROTX,#125252  ;CHECK DATA
012204  001402                          BEQ     TS174
012206                          ROT7:
012206  104000                          EMT                     ;ROL W/ MODE 7 FAILED
012210  000000                  ROTXAD: 0


                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 0 SWAB INSTRUCTION.  R0 IS SET TO
                                ;177400.  A SWAB MODE 0 IS EXECUTED AND THE CONDITIONAL BRANCH
                                ;IS USED TO CHECK THE SIGN OF THE RESULT.  ALSO, A COMPARISON
                                ;IS MADE TO CHECK THE DATA RESULTS.
                                ;
                                ;*****************************************************************************
                                ;TEST 174        TEST MODE 0 W/ SWAB INST.
                                ;*****************************************************************************
012212                          TS174:
012212  012700  177400                  MOV     #177400,R0      ;MOVE TEST PATTERN TO R0
012216  000300                          SWAB    R0              ;TRY SWAB MODE 0
012220  100401                          BMI     SB0
012222  104000                          EMT                     ;SWAB DID NOT SET CC'S CORRECT
012224  022700  000377          SB0:    CMP     #377,R0         ;CHECK RESULT
012230  001401                          BEQ     TS175
012232  104000                          EMT                     ;RESULT OF SWAB MODE 0 FAILED


                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES MODE 1 SWAB INSTRUCTION.  THE TEST
                                ;PATTERN IS MOVED TO LOC 0.  R0 IS CLEARED AND USED AS THE ADDRESSING
                                ;REGISTER IN THE MODE 1 SWAB.  THE DATA RESULTS ARE CHECKED WITH
                                ;A COMPARE.
                                ;
                                ;*****************************************************************************
                                ;TEST 175        TEST MODE 1 W/ SWAB INST
                                ;*****************************************************************************
012234                          TS175:
012234  012737  125652  000000          MOV     #125652,@#0     ;MOVE TEST PATTERN TO LOC. 0
012242  005000                          CLR     R0              ;R0=0
012244  000310                          SWAB    (R0)            ;TRY SWAB MODE 1
012246  022737  125253  000000          CMP     #125253,@#0     ;CHECK RESULT
012254  001401                          BEQ     TS176
012256  104000                          EMT                     ;RESULT OF SWAB MODE 1 FAILED
```

```
3844
3845
3846                    ;*****************************************************************************
3847                    ;
3848                    ;       THIS TEST VERIFIES MODE 2 SWAB INSTRUCTION.  THE TEST
3849                    ;PATTERN IS MOVED TO LOC 0.  R0 IS CLEARED AND USED AS THE MODE
3850                    ;2 ADDRESSING REGISTER.  THE RESULTS ARE CHECKED WITH A COMPARE.
3851                    ;R0 IS CHECKED FOR PROPER DECREMENTING.
3852                    ;
3853                    ;*****************************************************************************
3854                    ;TEST 176       TEST MODE 2 W/ SWAB INST
3855                    ;*****************************************************************************
3856 012260            TS176:
3857 012260 012737 125152 000000        MOV     #125152,@#0     ;MOVE TEST PATTERN TO LOC. 0
3858 012266 005000              CLR     R0              ;R0=0
3859 012270 000320              SWAB    (R0)+           ;TRY SWAB MODE 2
3860 012272 022737 065252 000000        CMP     #65252,@#0      ;CHECK RESULT
3861 012300 001401              BEQ     SB2
3862 012302 104000              EMT                     ;RESULT OF SWAB MODE 0 FAILED
3863 012304 162700 000002       SB2:    SUB     #2,R0           ;CHECK EFFECT OF REG.
3864 012310 001401              BEQ     TS177
3865 012312 104000              EMT                     ;REGISTER VALUE INCORRECT
3866
3867
3868                    ;*****************************************************************************
3869                    ;
3870                    ;       THIS TEST VERIFIES MODE 3 SWAB INSTRUCTION.  THE TEST
3871                    ;PATTERN IS MOVED TO LOC 0.  A MODE 3 SWAB INSTRUCTION IS EXECUTED
3872                    ;USING R7 AS THE ADDRESSING REGISTER.  A COMPARE VERIFIES THE
3873                    ;DATA RESULTS.
3874                    ;
3875                    ;*****************************************************************************
3876                    ;TEST 177       TEST MODE 3 W/SWAB INST.
3877                    ;*****************************************************************************
3878 012314            TS177:
3879 012314 012737 000377 000000        MOV     #377,@#0        ;MOVE TEST PATTERN TO LOC. 0
3880 012322 000337 000000       SWAB    @#0             ;TRY SWAB W/ MODE 3
3881 012326 022737 177400 000000        CMP     #177400,@#0     ;CHECK RESULT
3882 012334 001401              BEQ     TS200
3883 012336 104000              EMT                     ;RESULT OF SWAB INCORRECT
3884
3885
3886                    ;*****************************************************************************
3887                    ;
3888                    ;       THIS TEST VERIFIES MODE 4 SWAB INSTRUCTIONS.  THE DATA
3889                    ;IS MOVED TO LOC 0.  R0 IS SET TO 2 AND USED AS THE MODE 4 ADDRESSING
3890                    ;REGISTER.  THE DATA IS CHECKED WITH A COMPARE AND R0 IS CHECKED
3891                    ;FOR PROPER DECREMENTING.
3892                    ;
3893                    ;*****************************************************************************
3894                    ;TEST 200       TEST MODE 4 W/ SWAB INST
3895                    ;*****************************************************************************
3896 012340            TS200:
3897 012340 012737 125652 000000        MOV     #125652,@#0     ;MOVE TEST PATTERN TO LOC. 0
3898 012346 012700 000002       MOV     #2,R0           ;SET UP REGISTER POINTER
3899 012352 000340              SWAB    -(R0)           ;TRY SWAB MODE 4
```

```
3900  012354  022737  125253  000000          CMP     #125253,@#0     ;CHECK RESULT
3901  012362  001401                          BEQ     SB4
3902  012364  104000                          EMT                     ;RESULT OF SWAB INCORRECT
3903  012366  005700                  SB4:    TST     R0              ;CHECK EFFECT ON REG.
3904  012370  001401                          BEQ     TS201
3905  012372  104000                          EMT                     ;REGISTER VALUE INCORRECT
3906
3907
3908                                  ;*******************************************************************************
3909                                  ;
3910                                  ;       THIS TEST VERIFIES MODE 5 SWAB INSTRUCTION.  THE TEST USES
3911                                  ;TWO LOCATIONS FOLLOWING THE TEST CODE.  SB5X HOLDS THE DATA;
3912                                  ;SB5XAD IS A POINTER TO THE DATA LOCATION.  THE DATA IS MOVED TO
3913                                  ;SB5X AND R0 IS SET TO TWO PLUS THE ADDRESS OF SB5XAD.  FOLLOWING
3914                                  ;THE MODE 5 SWAB SB5X IS CHECKED FOR THE PROPER DATA.  R0 IS
3915                                  ;CHECKED TO SEE THAT IT WAS DECREMENTED PROPERLY.
3916                                  ;
3917                                  ;*******************************************************************************
3918                                  ;TEST 201       TEST MODE 5 W/ SWAB INST.
3919                                  ;*******************************************************************************
3920  012374                          TS201:
3921  012374  012700  012436                  MOV     #SB5XAD+2,R0    ;SET UP POINTER TO WORK LOCATION
3922  012400  012767  125125  000024          MOV     #125125,SB5X    ;MOVE PATTERN TO WORK LOCATION
3923  012406  000350                          SWAB    @-(R0)          ;TRY SWAB MODE 5
3924  012410  022767  052652  000014          CMP     #52652,SB5X     ;CHECK RESULT
3925  012416  001401                          BEQ     SB5A
3926  012420  104000                          EMT                     ;RESULT OF SWAB INCORRECT
3927  012422  020027  012434          SB5A:   CMP     R0,#SB5XAD      ;CHECK RESULT OF REG.
3928  012426  001403                          BEQ     TS202
3929  012430                          SB5:
3930  012430  104000                          EMT                     ;REGISTER VALUE INCORRECT
3931  012432  000000                  SB5X:   0                       ;WORK LOCATION
3932  012434  012432                  SB5XAD: SB5X
3933
3934
3935                                  ;*******************************************************************************
3936                                  ;
3937                                  ;       THIS TEST VERIFIES MODE 6 SWAB INSTRUCTION.  THIS TEST
3938                                  ;USES A WORK LOCATION (SB6X) FOLLOWING THE TEST CODE.  TEST DATA
3939                                  ;IS LOADED INTO THE WORK LOCATION.  R0, THE ADDRESSING REGISTER
3940                                  ;IS LOADED WITH 6 LESS THEN THE ADDRESS OF THE WORK LOCATION.
3941                                  ;THE MODE 6 SWAB IS EXECUTED WITH A +6 OFFSET.  THE DATA IS
3942                                  ;VERIFIED WITH A COMPARE.
3943                                  ;
3944                                  ;*******************************************************************************
3945                                  ;TEST 202       TEST MODE 6 W/ SWAB INST.
3946                                  ;*******************************************************************************
3947  012436                          TS202:
3948  012436  012767  125125  000022          MOV     #125125,SB6X    ;MOVE PATTERN TO WORK LOCATION
3949  012444  012700  012460                  MOV     #SB6X-6,R0      ;MOVE OFFSET POINTER TO R0
3950  012450  000360  000006                  SWAB    6(R0)           ;TRY SWAB W/ MODE 6
3951  012454  022760  052652  000006          CMP     #52652,6(R0)    ;CHECK RESULT
3952  012462  001402                          BEQ     TS203
3953  012464                          SB6:
3954  012464  104000                          EMT                     ;RESULT OF SWAB INCORRECT
3955  012466  000000                  SB6X:   0                       ;WORK LOCATION
```

```
3956
3957
3958                    ;*****************************************************************************************
3959                    ;
3960                    ;       THIS TEST VERIFIES MODE 7 SWAB INSTRUCTION.  THIS TEST
3961                    ;USES TWO LOCATIONS FOLLOWING THE TEST CODE:  A WORK LOCATION
3962                    ;(SB7X) AND A POINTER TO THE WORK LOCATION (SB7XAD).  DATA IS MOVED
3963                    ;TO THE WORK LOCATION.  R0 IS LOADED WITH 72 LESS THAN THE ADDRESS
3964                    ;OF THE ADDRESS POINTER.  THE DATA IS SWAB'ED USING A MODE 7
3965                    ;INSTRUCTION WITH AN OFFSET OF +72.  THE DATA IS VERIFIED WITH A
3966                    ;COMPARE.
3967                    ;
3968                    ;*****************************************************************************************
3969                    ;TEST 203       TEST MODE 7 W/ SWAB INST.
3970                    ;*****************************************************************************************
3971  012470            TS203:
3972  012470  012767  177400  000022            MOV     #177400,SB7X    ;MOVE PATTERN TO WORK LOCATION
3973  012476  012700  012430                    MOV     #SB7XAD-72,R0   ;MOVE OFFSET POINTER TO R0
3974  012502  000370  000072                    SWAB    @72(R0)         ;TRY SWAB MODE 7
3975  012506  027027  000072  000377            CMP     @72(R0),#377    ;CHECK RESULTS
3976  012514  001403                            BEQ     TS204
3977  012516            SB7:
3978  012516  104000                            EMT                     ;RESULT OF SWAB INCORRECT
3979  012520  000000    SB7X:   0               ;WORK LOCATION
3980  012522  012520    SB7XAD: SB7X            ;POINTER TO WORK LOCATION
3981
3982
3983                    ;*****************************************************************************************
3984                    ;
3985                    ;       THIS TEST VERIFIES ALL LEGAL MODES OF THE JMP INSTRUCTION.
3986                    ;BECAUSE OF THE NATURE OF THE INSTRUCTION UNDER TEST, THIS TEST
3987                    ;UTILIZES SEVERAL DIFFERENT TECHNIQUES.  THE CODE IS NOT EXECUTED
3988                    ;IN A LINEAR FASHION.  THE DIFFERENT MODES ARE EXECUTED IN ORDER
3989                    ;FROM 1-7; HOWEVER, THE CODE IS ARRANGED SO THAT CONTROL LEAP
3990                    ;FROGS THRU THE TEST CODE.  THE ORDER OF APPEARANCE OF THE CODE
3991                    ;IS:
3992                    ;       JMP MODE 1
3993                    ;       JMP MODE 3
3994                    ;       JMP MODE 2
3995                    ;       JMP MODE 4
3996                    ;       JMP MODE 6
3997                    ;       JMP MODE 5
3998                    ;       JMP MODE 7
3999                    ;AN INTERNAL SEQUENCE TEST (JMPSEQ) IS USED TO INSURE THAT THE
4000                    ;JUMPS ARE OCCURRING IN THE PROGRAMMED SEQUENCE.
4001                    ;       THE TEST IS MADE UP OF SEVERAL BLOCKS OF CODE.  EACH CODE
4002                    ;BEGINS WITH A LABEL WHICH INDICATES THE MODE BEING EXECUTED IN
4003                    ;THAT BLOCK.  A SIMPLE PROCEDURE IS FOLLOWED IN EACH BLOCK.  FOR
4004                    ;EXAMPLE THE CODE BEGINNING AT JMP3 WILL FIRST COMPARE THE RESULTS
4005                    ;OF THE PREVIOUS MODE 2 JUMP.  (ANY REGISTER CHANGES ARE VERIFIED
4006                    ;AND THE SEQUENCE CHECK IS MADE).  THEN THE REGISTERS ARE SETUP
4007                    ;FOR A MODE 3 JUMP TO THE NEXT TEST BLOCK (HERE, JMP4), THE SEQUENCE
4008                    ;CHECKER IS UPDATED AND THE JUMP IS EXECUTED.
4009                    ;       IF A FAILURE OCCURS, THE SEQUENCE CHECKER WILL ASSIST IN
4010                    ;DETERMINING JUST WHICH MODE FAILED.  IF THE SEQUENCE IS CORRECT
4011                    ;THEN THE ERROR DETECTED WAS A MODE FAILURE (E.G. FAILURE OF THE
```

```
                                        ;REGISTER TO BE INCREMENTED IN MODE 2 JUMP.)
                                        ;
                                        ;*****************************************************************************
                                        ;TEST 204       TEST THE JMP INSTRUCTION IN ALL MODES
                                        ;*****************************************************************************
012524                                  TS204:
012524  005067  000240                          CLR     JMPSEQ          ;ESTABLISH A SEQUENCE CHECKER
012530  012700  012574                          MOV     #JMP2,R0        ;SET R0=JUMP TARGET
012534  000110                                  JMP     (R0)            ;TRY JMP MODE 1
012536  022700  012540                  JMP3:   CMP     #.+2,R0         ;CHECK RESULT OF MODE 2 JUMP
012542  001401                                  BEQ     JMP3A
012544  104000                                  EMT                     ;REGISTER VALUE AFTER JMP MODE 2 INCORRECT
012546  026727  000216  000001        JMP3A:  CMP     JMPSEQ,#1       ;MAKE SURE JMPS ARE IN SEQUENCE: JMPSEQ=1?
012554  001401                                  BEQ     JMP3B
012556  104000                                  EMT                     ;SHOULD BE HERE FROM JMP MODE 2 ONLY
012560  012700  012572                  JMP3B:  MOV     #IJMP4,R0       ;POINT R0 TO INDIRECT JMP ADDR.
012564  005267  000200                          INC     JMPSEQ          ;UPDATE SEQUENCE CHECKER
012570  000130                                  JMP     @(R0)+          ;TRY JMP MODE 3
012572  012616                  IJMP4:  JMP4                            ;ADDRESS INDIRECT JUMP
                                        
012574  005767  000170                  JMP2:   TST     JMPSEQ          ;CHECK THAT JMPS ARE IN SEQUENCE: JMPSEQ=0?
012600  001401                                  BEQ     JMP2A
012602  104000                                  EMT                     ;SHOULD BE HERE FROM JMP MODE 1 ONLY
012604  005267  000160                  JMP2A:  INC     JMPSEQ          ;UPDATE SEQUENCE CHECKER
012610  012700  012536                          MOV     #JMP3,R0        ;SET R0=JUMP TARGET
012614  000120                                  JMP     (R0)+           ;TRY A JUMP MODE 2 TO ''JMP3''
012616  022700  012574                  JMP4:   CMP     #IJMP4+2,R0     ;CHECK RESULT OF REGISTER IN MODE 3 JUMP
012622  001401                                  BEQ     JMP4A
012624  104000                                  EMT                     ;REGISTER VALUE AFTER MODE 3 JUMP INCORRECT
012626  022767  000002  000134        JMP4A:  CMP     #2,JMPSEQ       ;CHECK JUMP SEQUENCE: JMPSEQ=2?
012634  001401                                  BEQ     JMP4B
012636  104000                                  EMT                     ;SHOULD BE ONLY FROM MODE 3 JUMP
012640  012700  012702                  JMP4B:  MOV     #JMP5+2,R0      ;SET UP POINTER TO JUMP TARGET
012644  005267  000120                          INC     JMPSEQ          ;UPDATE SEQUENCE CHECKER
012650  000140                                  JMP     -(R0)           ;TRY JUMP MODE 4 TO ''JMP4''
                                        
012652  022767  000004  000110        JMP6:   CMP     #4,JMPSEQ       ;CHECK THAT JUMPS ARE IN SEQUENCE: JMPSEQ=4?
012660  001401                                  BEQ     JMP6A
012662  104000                                  EMT                     ;SHOULD BE HERE ONLY FROM MODE 5 JUMP
012664  012700  013324                  JMP6A:  MOV     #JMP7+376,R0    ;SET UP OFFSET POINTER TO JUMP TARGET
012670  005267  000074                          INC     JMPSEQ          ;UPDATE JUMP SEQUENCE
012674  000160  177402                          JMP     -376(R0)        ;TRY MODE 6 JUMP
                                        
012700  022767  000003  000062        JMP5:   CMP     #3,JMPSEQ       ;CHECK THAT JUMPS ARE IN SEQUENCE: JMPSEQ=3?
012706  001401                                  BEQ     JMP5A
012710  104000                                  EMT                     ;SHOULD ONLY BE HERE FROM MODE 4 JUMP
012712  012700  012726                  JMP5A:  MOV     #IJMP5+2,R0     ;SET UP POINTER TO INDIRECT JUMP ADDR.
012716  005267  000046                          INC     JMPSEQ          ;UPDATE JUMP SEQUENCE
012722  000150                                  JMP     @-(R0)          ;TRY JUMP MODE 5 TO ''JMP6''
012724  012652                  IJMP5:  JMP6                            ;INDIRECT ADDRESS POINTER
                                        
012726  022767  000005  000034        JMP7:   CMP     #5,JMPSEQ       ;CHECK JUMPS IN SEQUENCE: JMPSEQ=5?
012734  001401                                  BEQ     JMP7A
012736  104000                                  EMT                     ;SHOULD ONLY BE HERE FROM MODE 6 JUMP
012740  012700  012764                  JMP7A:  MOV     #IJMP+10,R0     ;SET UP OFFSET POINTER TO INDIRECT ADDR.
012744  005267  000020                          INC     JMPSEQ          ;UPDATE JUMP SEQUENCE
```

```
4068  012750  000170  177770            JMP     @-10(R0)        ;TRY MODE 7 JUMP
4069  012754  012756                    IJMP:   JMPCK           ;INDIRECT ADDRESS
4070
4071  012756  026727  000006  000006    JMPCK:  CMP     JMPSEQ,#6       ;CHECK JUMPS IN SEQUENCE: JMPSEQ
4072  012764  001402                            BEQ     TS205
4073  012766  104000                            EMT                     ;SHOULD ONLY BE HERE FROM MODE 6 JUMP
4074  012770  000000                    JMPSEQ: 0
4075
4076                                    ;*****************************************************************************
4077                                    ;
4078                                    ;       THIS TEST VERIFIES ALL LEGAL MODES OF THE JSR INSTRUCTION.
4079                                    ;THE CONCEPT OF LEAP FROGGING AND SEQUENCE CHECKING (JSRSEQ) IS
4080                                    ;IDENTICAL TO THAT USED IN JMP TEST (SEE PREVIOUS TEST).  EACH
4081                                    ;BLOCK OF CODE VERIFIES THE PREVIOUS JSR BY CHECKING THE SEQUENCE,
4082                                    ;CHECKING THAT THE PC WAS SAVED IN THE SPECIFIED REGISTER, CHECKING
4083                                    ;THAT THE SP WAS DECREMENTED, CHECKING THAT THE REGISTER WAS
4084                                    ;SAVED ON THE STACK, AND FINALLY CHECKING THAT ANY MODE ADDRESS
4085                                    ;REGISTER ALTERATIONS (E.G. INCREMENT REGISTER IN MODE 2) WERE
4086                                    ;SUCCESSFUL.  R1 IS USED AS THE REGISTER IN ALL JSR INSTRUCTIONS.
4087                                    ;       IF A FAILURE OCCURS, THE SEQUENCE CHECKER WILL ASSIST IN
4088                                    ;DETERMINING JUST WHICH MODE FAILED.  IF THE SEQUENCE IS CORRECT
4089                                    ;THEN THE ERROR DETECTED WAS A FUNCTIONAL FAILURE (E.G., INCORRECT
4090                                    ;REGISTER SAVED).
4091                                    ;
4092                                    ;*****************************************************************************
4093                                    ;TEST 205       TEST JSR INSTRUCTION W/ ALL MODES
4094                                    ;*****************************************************************************
4095  012772                            TS205:
4096  012772  000402                            BR      JSR1
4097  012774  000137  013356            JSR0:   JMP     @#JSRCK1
4098
4099  013000  012706  001000            JSR1:   MOV     #STBOT,R6       ;SET STACK POINTER
4100  013004  012700  013076                    MOV     #JSR2,R0        ;SET TARGET ADDRESS
4101  013010  005037  013336                    CLR     @#JSRSEQ        ;INITIALIZE SEQUENCE CHECKER
4102  013014  005001                            CLR     R1              ;INITIALIZE R1
4103  013016  005101                            COM     R1
4104  013020  004110                            JSR     R1,(R0)         ;TRY JSR MODE 1
4105                                                                    ; TO SCOPE: REPLACE THE MOVE INSTRUCTION   <====
4106                                                                    ;           FOLLOWING W/ 774               <====
4107  013022                            JSR1A:
4108  013022  104000                            EMT                     ;JSR MODE 1 FAILED
4109
4110  013024  022737  000001  013336    JSR3:   CMP     #1,@# JSRSEQ    ;CHECK SEQUENCE: JSRSEQ=1?
4111  013032  001014                            BNE     JSR3A           ;BRANCH IF OUT OF SEQUENCE
4112  013034  020127  013152                    CMP     R1,#JSR4        ;PROPER PC SAVED?
4113  013040  001011                            BNE     JSR3A           ;BRANCH IF PC WRONG
4114  013042  022706  000776                    CMP     #STBOT-2,R6     ;STACK POINTER DECREMENTED?
4115  013046  001006                            BNE     JSR3A           ;BRANCH IF SP WRONG
4116  013050  022716  125252                    CMP     #125252,(R6)    ;REG SAVED ON STACK?
4117  013054  001003                            BNE     JSR3A           ;BRANCH IF REG. NOT SAVED
4118  013056  022700  013026                    CMP     #JSR3+2,R0      ;MODE 2 INCREMENT CORRECT?
4119  013062  001401                            BEQ     JSR3B
4120  013064                            JSR3A:
4121  013064  104000                            EMT                     ;JSR MODE 3 MALFUNCTIONED
4122  013066  005237  013336            JSR3B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
4123  013072  004137  013152                    JSR     R1,@#JSR4       ;TRY JSR MODE 4
```

```
4124
4125  013076  005737  013336          JSR2:   TST     @#JSRSEQ        ;CHECK SEQUENCE: JSRSEQ=0?
4126  013102  001011                          BNE     JSR2A           ;BRANCH IF OUT OF SEQUENCE
4127  013104  020127  013022                  CMP     R1,#JSR1A       ;PROPER PC SAVED?
4128  013110  001006                          BNE     JSR2A           ;BRANCH IF PC WRONG
4129  013112  022706  000776                  CMP     #STBOT-2,R6     ;R6 DECREMENT?
4130  013116  001003                          BNE     JSR2A           ;BRANCH IF R6 IS INCORRECT
4131  013120  021627  177777                  CMP     (R6),#-1        ;REGISTER SAVED?
4132  013124  001401                          BEQ     JSR2B
4133  013126                          JSR2A:
4134  013126  104000                          EMT                     ;JSR MODE 1 MALFUNCTIONED
4135  013130  012706  001000          JSR2B:  MOV     #STBOT,R6       ;INITIALIZE R6
4136  013134  012701  125252                  MOV     #125252,R1      ;INITIALIZE R1
4137  013140  005237  013336                  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
4138  013144  012700  013024                  MOV     #JSR3,R0        ;SET TARGET ADDRESS
4139  013150  004120                          JSR     R1,(R0)+        ;TRY JSR MODE 2
4140
4141  013152  022737  000002  013336  JSR4:   CMP     #2,@#JSRSEQ     ;CHECK SEQUENCE: JSRSEQ=2?
4142  013160  001003                          BNE     JSR4A           ;BRANCH IF OUT OF SEQUENCE
4143  013162  022701  013076                  CMP     #JSR2,R1        ;PROPER PC SAVED?
4144  013166  001401                          BEQ     JSR4B
4145  013170                          JSR4A:
4146  013170  104000                          EMT                     ;JSR MODE 3 MALFUNCTIONED
4147  013172  005237  013336          JSR4B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
4148  013176  012700  013244                  MOV     #JSR5+2,R0      ;SET TARGET ADDRESS
4149  013202  004140                          JSR     R1,-(R0)        ;TRY JSR MODE 4
4150
4151  013204  022767  000004  000124  JSR6:   CMP     #4,JSRSEQ       ;CHECK SEQUENCE: JSRSEQ=4?
4152  013212  001006                          BNE     JSR6A           ;BRANCH IF OUT OF SEQUENCE
4153  013214  022701  013302                  CMP     #JSR7,R1        ;PROPER PC SAVED?
4154  013220  001003                          BNE     JSR6A           ;BRANCH IF PC WRONG
4155  013222  022700  013332                  CMP     #JSR6AD,R0      ;MODE 5 REGISTER CORRECT?
4156  013226  001401                          BEQ     JSR6B
4157  013230                          JSR6A:
4158  013230  104000                          EMT                     ;JSR MODE 5 FAILED
4159  013232  005237  013336          JSR6B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
4160  013236  004167  000040                  JSR     R1,JSR7         ;TRY JSR MODE 6
4161  013242  022767  000003  000066  JSR5:   CMP     #3,JSRSEQ       ;CHECK SEQUENCE: JSRSEQ=3?
4162  013250  001006                          BNE     JSR5A           ;BRANCH IF OUT OF SEQUENCE
4163  013252  022701  013204                  CMP     #JSR6,R1        ;PROPER PC SAVED?
4164  013256  001003                          BNE     JSR5A           ;BRANCH IF PC WRONG
4165  013260  022700  013242                  CMP     #JSR5,R0        ;CHECK MODE 4 REGISTER
4166  013264  001401                          BEQ     JSR5B
4167  013266                          JSR5A:
4168  013266  104000                          EMT                     ;JSR MODE 4 MALFUNCTIONED
4169  013270  005237  013336          JSR5B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
4170  013274  012700  013334                  MOV     #JSR6AD+2,R0    ;POINT R0 TO TARGET ADDRESS
4171  013300  004150                          JSR     R1,@-(R0)       ;TRY JSR MODE 5
4172
4173  013302  022737  000005  013336  JSR7:   CMP     #5,@#JSRSEQ     ;CHECK SEQUENCE: JSRSEQ=5?
4174  013310  001003                          BNE     JSR7A           ;BRANCH IF OUT OF SEQUENCE
4175  013312  022701  013242                  CMP     #JSR5,R1        ;PROPER PC SAVED?
4176  013316  001401                          BEQ     JSR7B
4177  013320                          JSR7A:
4178  013320  104000                          EMT                     ;JSR MODE 6 FAILED
4179  013322  005237  013336          JSR7B:  INC     @#JSRSEQ        ;UPDATE SEQUENCE CHECKER
```

```
4180  013326  004177  000002          JSR     R1,@JSRCKAD     ;TRY JSR MODE 7
4181
4182  013332  013204                  JSR6AD: JSR6                    ;MODE 5 TARGET ADDRESS
4183  013334  013340                  JSRCKAD:JSRCK                   ;MODE 7 TARGET ADDRESS
4184  013336  000000                  JSRSEQ: 0                       ;SEQUENCE CHECKER
4185
4186  013340  022767  000006  177770 JSRCK:  CMP     #6,JSRSEQ       ;CHECK SEQUENCE: JSRSEQ=6?
4187  013346  001003                          BNE     JSRCK1          ;BRANCH IF OUT OF SEQUENCE
4188  013350  022701  013332                  CMP     #JSR6AD,R1      ;PROPER PC SAVED?
4189  013354  001401                          BEQ     TS206
4190  013356                          JSRCK1:
4191  013356  104000                          EMT                     ;JSR MODE 7 MALFUNCTIONED
4192
4193
4194                                  ;*******************************************************************************
4195                                  ;
4196                                  ;       THIS TEST VERIFIES THE RTS INSTRUCTION.  THE STACK POINTER
4197                                  ;IS INITIALIZED AND A TEST PATTERN STORED ON STACK.  R0 IS LOADED
4198                                  ;WITH RETURN ADDRESS.  AN RTS IS EXECUTED, AND, AT THE TARGET
4199                                  ;ADDRESS, A CHECK IS MADE THAT R0 WAS PROPERLY RESTORED FROM THE
4200                                  ;STACK.
4201                                  ;*******************************************************************************
4202                                  ;TEST 206       TEST RTS INSTRUCTION
4203                                  ;*******************************************************************************
4204  013360                          TS206:
4205  013360  012706  001000                  MOV     #STBOT,R6       ;INITIALIZE STACK POINTER
4206  013364  012746  052525                  MOV     #52525,-(R6)    ;INITIALIZE TOP OF STACK
4207  013370  012700  013400                  MOV     #RTS1,R0        ;INITIALIZE RETURN REGISTER
4208  013374  000200                          RTS     R0              ;TRY RTS THROUGH R0
4209                                                                  ; TO SCOPE: REPLACE THE MOVE INSTRUCTION   <====
4210                                                                  ;           FOLLOWING W/ 770               <====
4211  013376  104000                          EMT                     ;RTS FAILED
4212  013400  022700  052525          RTS1:   CMP     #52525,R0       ;CHECK THAT R0 RESTORED FROM STACK
4213  013404  001401                          BEQ     TS207
4214  013406  104000                          EMT                     ;RTS MALFUNCTIONED
4215                                  ;*******************************************************************************
4216                                  ;
4217                                  ;       THESE NEXT FOUR TESTS VERIFY THE FUNCTIONING OF A GROUP
4218                                  ;OF FOUR INSTRUCTIONS.  THE GROUP CONSISTS OF THE INSTRUCTIONS:
4219                                  ;MOV, BIC, BIT, AND BIS.  THESE INSTRUCTIONS ARE SIMILAR IN THE
4220                                  ;WAY THEY EFFECT THE C AND V BITS.  THEY ALL LEAVE THE V-BIT
4221                                  ;CLEAR AND THE C-BIT UNAFFECTED.
4222                                  ;       THE TEST PROCEDURE IS AS FOLLOWS:  THE N, Z, AND V BITS
4223                                  ;ARE LOADED WITH THE COMPLEMENT OF THE EXPECTED RESULTS, THE C-BIT
4224                                  ;IS LOADED WITH THE DESIRED RESULT.  THE INSTRUCTION IS EXECUTED
4225                                  ;WITH DIFFERENT DATA PATTERNS AND THE RESULTS ARE VERIFIED WITH
4226                                  ;A SERIES OF CONDITIONAL BRANCH INSTRUCTIONS.  THE DATA IS CHOSEN
4227                                  ;TO PRODUCT ALL POSSIBLE COMBINATIONS OF THE C AND V BITS.
4228                                  ;
4229                                  ;*******************************************************************************
4230                                  ;TEST 207       TEST MOV INSTRUCTION
4231                                  ;*******************************************************************************
4232  013410                          TS207:
4233  013410  000277                          SCC                     ;CC=0110
4234  013412  000251                          +CLN!CLC
4235  013414  012700  100000                  MOV     #100000,R0      ;CC=1000
```

```
4236  013420  101402                BLOS    MOV1
4237  013422  102401                BVS     MOV1
4238  013424  100401                BMI     MOV2
4239  013426                    MOV1:
4240  013426  104000                EMT                     ;MOV DID NOT SET CC'S CORRECTLY
4241
4242  013430  000277            MOV2: SCC                     ;CC=1011
4243  013432  000244                CLZ
4244  013434  012700  000000        MOV     #0,R0           ;CC=0101
4245  013440  101002                BHI     MOV3            ;C OR Z = 0?
4246  013442  102401                BVS     MOV3            ;V=1?
4247  013444  100001                BPL     TS210
4248  013446                    MOV3:
4249  013446  104000                EMT                     ;MOV DID NOT SET CC'S CORRECTLY
4250                        ;*****************************************************************************
4251                        ;TEST 210       TEST BIT INSTRUCTION
4252                        ;*****************************************************************************
4253  013450                TS210:
4254  013450  012700  100001        MOV     #100001,R0
4255  013454  000277                SCC                     ;CC=0110
4256  013456  000251                +CLN!CLC
4257  013460  032700  100000        BIT     #100000,R0      ;CC=1000
4258  013464  101402                BLOS    BITST1
4259  013466  102401                BVS     BITST1
4260  013470  100401                BMI     BITST2
4261  013472                    BITST1:
4262  013472  104000                EMT                     ;BIT DID NOT SET CC'S CORRECTLY
4263
4264  013474  000277            BITST2: SCC                 ;CC=1011
4265  013476  000244                CLZ
4266  013500  032700  077776        BIT     #77776,R0       ;CC=0101
4267  013504  101002                BHI     BITST3
4268  013506  102401                BVS     BITST3
4269  013510  100001                BPL     TS211
4270  013512                    BITST3:
4271  013512  104000                EMT                     ;BIT DID NOT SET CC'S CORRECTLY
4272                        ;*****************************************************************************
4273                        ;TEST 211       TEST BIC INSTRUCTION
4274                        ;*****************************************************************************
4275  013514                TS211:
4276  013514  012700  177777        MOV     #177777,R0
4277  013520  000277                SCC                     ;CC=0110
4278  013522  000251                +CLN!CLC
4279  013524  042700  077777        BIC     #77777,R0       ;CC=1000
4280  013530  101402                BLOS    BIC1
4281  013532  102401                BVS     BIC1
4282  013534  100401                BMI     BIC2
4283  013536                    BIC1:
4284  013536  104000                EMT                     ;BIC DID NOT SET CC'S CORRECTLY
4285  013540  000277            BIC2:   SCC                 ;CC=1011
4286  013542  000244                CLZ
4287  013544  042700  100000        BIC     #100000,R0      ;CC=0101
4288  013550  101002                BHI     BIC3
4289  013552  102401                BVS     BIC3
4290  013554  100001                BPL     TS212
4291  013556                    BIC3:
```

```
4292  013556  104000                        EMT                     ;BIC DID NOT SET CC'S CORRECTLY
4293                                ;*****************************************************************************
4294                                ;TEST 212       TEST BIS INSTRUCTION
4295                                ;*****************************************************************************
4296  013560                        TS212:
4297  013560  005000                        CLR     R0              ;R0=0
4298  013562  000277                        SCC                     ;CC=1010
4299  013564  000251                        +CLN!CLC
4300  013566  052700  000000              BIS     #0,R0           ;CC=0100  R0=0
4301  013572  103403                        BCS     BIS1
4302  013574  102402                        BVS     BIS1
4303  013576  100401                        BMI     BIS1
4304  013600  001401                        BEQ     BIS2
4305  013602                        BIS1:
4306  013602  104000                        EMT                     ;BIS DID NOT SET CC'S CORRECTLY
4307  013604  000277                BIS2:   SCC                     ;CC=0111
4308  013606  000250                        CLN
4309  013610  052700  177777              BIS     #177777,R0      ;CC=1001
4310  013614  103003                        BCC     BIS3
4311  013616  102402                        BVS     BIS3
4312  013620  001401                        BEQ     BIS3
4313  013622  100401                        BMI     TS213
4314  013624                        BIS3:
4315  013624  104000                        EMT                     ;BIS DID NOT SET CC'S CORRECTLY
4316
4317                                ;*****************************************************************************
4318                                ;
4319                                ;       THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE INC AND
4320                                ;DEC INSTRUCTIONS.  THESE INSTRUCTIONS BOTH EFFECT THE C AND V
4321                                ;BITS THE SAME; THE C-BIT IS LEFT UNCHANGED AND THE V-BIT IS DEPENDENT
4322                                ;UPON THE DATA RESULTS.  THE SAME PROCEDURE IS USED.  THE CONDITION
4323                                ;CODE BITS ARE INITIALIZED, THE INSTRUCTION IS EXECUTED AND THE
4324                                ;RESULTS ARE VERIFIED WITH A SERIES OF CONDITIONAL BRANCH INSTRUCTIONS.
4325                                ;THIS PROCEDURE IS REPEATED WITH SEVERAL DATA PATTERNS TO PRODUCE
4326                                ;DIFFERENT COMBINATIONS OF THE C AND V BITS.
4327                                ;
4328                                ;*****************************************************************************
4329                                ;TEST 213       TEST INC INSTRUCTION
4330                                ;*****************************************************************************
4331  013626                        TS213:
4332  013626  012700  077777              MOV     #077777,R0      ;R0=077777
4333  013632  000257                        CCC                     ;CC=0100
4334  013634  000264                        SEZ
4335  013636  005200                        INC     R0              ;CC=1010   R0=10000
4336  013640  101402                        BLOS    INC1
4337  013642  100001                        BPL     INC1
4338  013644  102401                        BVS     INC2
4339  013646                        INC1:
4340  013646  104000                        EMT                     ;INC DID NOT SET CC'S CORRECTLY
4341  013650  052700  077777        INC2:   BIS     #77777,R0       ;R0=177777
4342  013654  000261                        SEC                     ;CC=1011
4343  013656  000244                        CLZ
4344  013660  005200                        INC     R0              ;CC=0101   R0=0
4345  013662  100403                        BMI     INC3
4346  013664  102402                        BVS     INC3
4347  013666  103001                        BCC     INC3
```

```
4348  013670  001401                        BEQ     INC4
4349  013672                        INC3:
4350  013672  104000                        EMT                     ;INC DID NOT SET CC'S CORRECTLY
4351
4352  013674  000277                INC4:   SCC                     ;CC=1110
4353  013676  000241                        CLC
4354  013700  005200                        INC     R0              ;CC=0000   R0=1
4355  013702  101402                        BLOS    INC5
4356  013704  100401                        BMI     INC5
4357  013706  100001                        BPL     TS214
4358  013710                        INC5:
4359  013710  104000                        EMT                     ;INC DID NOT SET CC'S CORRECTLY
4360
4361                                ;*******************************************************************************
4362                                ;TEST 214       TEST DEC INSTRUCTION
4363                                ;*******************************************************************************
4364  013712                        TS214:
4365  013712  012700  000002               MOV     #2,R0           ;R0=2
4366  013716  000277                        SCC                     ;CC=1111
4367  013720  005300                        DEC     R0              ;CC=0001   R0=1
4368  013722  100403                        BMI     DEC1
4369  013724  001402                        BEQ     DEC1
4370  013726  102401                        BVS     DEC1
4371  013730  103401                        BCS     DEC2
4372  013732                        DEC1:
4373  013732  104000                        EMT                     ;DEC DID NOT SET CC'S CORRECTLY
4374  013734  000261                DEC2:   SEC                     ;CC=1011
4375  013736  000244                        CLZ
4376  013740  005300                        DEC     R0              ;CC=0101   R0=0
4377  013742  101002                        BHI     DEC3
4378  013744  100401                        BMI     DEC3
4379  013746  102001                        BVC     DEC4
4380  013750                        DEC3:
4381  013750  104000                        EMT                     ;DEC DID NOT SET CC'S CORRECTLY
4382  013752  000277                DEC4:   SCC                     ;CC=0110
4383  013754  000251                        +CLN!CLC
4384  013756  005300                        DEC     R0              ;CC=1000   R0=177777
4385  013760  101402                        BLOS    DEC5
4386  013762  102401                        BVS     DEC5
4387  013764  100401                        BMI     DEC6
4388  013766                        DEC5:
4389  013766  104000                        EMT                     ;DEC DID NOT SET CC'S CORRECTLY
4390  013770  042700  077777        DEC6:   BIC     #77777,R0       ;R0=100000
4391  013774  000277                        SCC                     ;CC=0101
4392  013776  000252                        +CLN!CLV
4393  014000  005300                        DEC     R0              ;CC=1011   R0=77777
4394  014002  100403                        BMI     DEC7            ;CC=0011
4395  014004  001402                        BEQ     DEC7
4396  014006  102001                        BVC     DEC7
4397  014010  103401                        BCS     TS215
4398  014012                        DEC7:
4399  014012  104000                        EMT                     ;DEC DID NOT SET CC'S CORRECTLY
4400
4401
4402                                ;*******************************************************************************
4403                                ;
```

```
                        ;       THESE NEXT THREE TESTS VERIFY THE FUNCTIONING OF THE CLR,
                        ;TST, AND SWAB INSTRUCTIONS.  THESE THREE INSTRUCTIONS ALL LEAVE
                        ;THE C AND V BITS CLEARED.  AGAIN, THE CONDITION CODES ARE PRESET,
                        ;THE INSTRUCTION EXECUTED AND THE RESULTS CHECKED WITH CONDITIONAL
                        ;BRANCH INSTRUCTIONS.  THE PROCEDURE IS REPEATED TO PRODUCE OTHER
                        ;COMBINATIONS OF CONDITION CODES.
                        ;
                        ;*****************************************************************************************
                        ;TEST 215       TEST CLR INSTRUCTION
                        ;*****************************************************************************************
014014                  TS215:
014014  000277                  SCC                     ;CC=1011
014016  000244                  CLZ
014020  005000                  CLR     R0              ;CC=0100   R0=0
014022  100403                  BMI     CLR1
014024  102402                  BVS     CLR1
014026  103401                  BCS     CLR1
014030  001401                  BEQ     TS216
014032                  CLR1:
014032  104000                  EMT                     ;CLR DID NOT SET CC'S CORRECTLY
                        ;*****************************************************************************************
                        ;TEST 216       TEST TST INSTRUCTION
                        ;*****************************************************************************************
014034                  TS216:
014034  000277                  SCC                     ;CC=1011
014036  000244                  CLZ
014040  005700                  TST     R0              ;CC=0100
014042  100403                  BMI     TEST1
014044  102402                  BVS     TEST1
014046  103401                  BCS     TEST1
014050  001401                  BEQ     TEST2
014052                  TEST1:
014052  104000                  EMT                     ;TEST DID NOT SET CC'S CORRECTLY
014054  005300          TEST2:  DEC     R0              ;MAKE R0 NEGATIVE
014056  000277                  SCC                     ;CC=0111
014060  000250                  CLN
014062  005700                  TST     R0              ;CC=1000
014064  101402                  BLOS    TEST3
014066  102401                  BVS     TEST3
014070  100401                  BMI     TS217
014072                  TEST3:
014072  104000                  EMT                     ;TEST DID NOT SET CC'S CORRECTLY
                        ;*****************************************************************************************
                        ;TEST 217       TEST SWAB INSTRUCTION
                        ;*****************************************************************************************
014074                  TS217:
014074  012700  170000          MOV     #170000,R0      ;R0=170000
014100  000277                  SCC                     ;CC=0111
014102  000250                  CLN
014104  000300                  SWAB    R0              ;CC=1000   R0=360
014106  101402                  BLOS    SWB1
014110  102401                  BVS     SWB1
014112  100401                  BMI     SWB2
014114                  SWB1:
014114  104000                  EMT                     ;SWAB DID NOT SET CC'S CORRECTLY
```

```
4460  014116  000277
4461  014120  000244
4462  014122  000300
4463  014124  102403
4464  014126  103402
4465  014130  100401
4466  014132  001401
4467  014134
4468  014134  104000
4469
4470
4471
4472
4473
4474
4475
4476
4477
4478
4479
4480
4481
4482
4483  014136
4484  014136  012700  040000
4485  014142  000277
4486  014144  062700  030000
4487  014150  101402
4488  014152  102401
4489  014154  100001
4490  014156
4491  014156  104000
4492  014160  000264
4493
4494  014162  062700  010000
4495  014166  101402
4496  014170  102001
4497  014172  100401
4498  014174
4499  014174  104000
4500  014176  000257
4501  014200  000270
4502  014202  062700  100000
4503  014206  101002
4504  014210  102001
4505  014212  100001
4506  014214
4507  014214  104000
4508  014216  062700  177777
4509  014222  101402
4510  014224  102401
4511  014226  100401
4512  014230
4513  014230  104000
4514  014232  000277
4515  014234  000245
```

```
SWB2:   SCC                     ;CC=1011
        CLZ
        SWAB    R0              ;CC=0100   R0=170000
        BVS     SWB3
        BCS     SWB3
        BMI     SWB3
        BEQ     TS220
SWB3:
        EMT                     ;

;*****************************************************************************************
;
;       THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE ADD AND
;ADC INSTRUCTIONS.  BOTH OF THESE INSTRUCTIONS HANDLE THE C AND
;V BITS IDENTICALLY.  THE PROCEDURE IS TO PRESET THE CONDITION
;CODES, EXECUTE THE INSTRUCTION WITH A PARTICULAR SET OF DATA, AND
;THEN CHECK THE RESULTS BY EXECUTING A SERIES OF CONDITIONAL
;BRANCHES.  THES PROCEDURE IS REPEATED SEVERAL TIMES WITH DIFFERENT
;DATA TO PRODUCE EVERY COMBINATION OF C AND V BITS.
;
;*****************************************************************************************
;TEST 220        TEST ADD INSTRUCTION
;*****************************************************************************************
TS220:
        MOV     #40000,R0       ;R0=40000
        SCC                     ;CC=1111
        ADD     #30000,R0       ;CC=0000   R0=70000
        BLOS    ADD1
        BVS     ADD1
        BPL     ADD2
ADD1:
        EMT                     ;ADD DID NOT SET CC'S CORRECTLY
ADD2:   SEZ                     ;CC=0100
        ADD     #10000,R0       ;CC=1010   40=100000
        BLOS    ADD3
        BVC     ADD3
        BMI     ADD4
ADD3:
        EMT                     ;ADD DID NOT SET CC'S CORRECTLY
ADD4:   CCC                     ;CC=1000
        SEN
        ADD     #100000,R0      ;CC=0111   R0=0
        BHI     ADD5
        BVC     ADD5
        BPL     ADD6
ADD5:
        EMT                     ;ADD DID NOT SET CC'S CORRECTLY
ADD6:   ADD     #177777,R0      ;CC=1000   R0=177777
        BLOS    ADD7
        BVS     ADD7
        BMI     ADD8
ADD7:
        EMT                     ;ADD DID NOT SET CC'S CORRECTLY
ADD8:   SCC                     ;CC=1010
        +CLC!CLZ
```

```
4516  014236  062700  000001          ADD     #1,R0           ;CC=0101   R=0
4517  014242  102403                  BVS     ADD9
4518  014244  103002                  BCC     ADD9
4519  014246  100401                  BMI     ADD9
4520  014250  001401                  BEQ     TS221
4521  014252                  ADD9:
4522  014252  104000                  EMT                     ;ADD DID NOT SET CC'S CORRECTLY
4523
4524          ;*****************************************************************************************
4525          ;TEST 221       TEST ADC INSTRUCTION
4526          ;*****************************************************************************************
4527  014254  TS221:
4528  014254  012700  077777          MOV     #077777,R0
4529  014260  000277                  SCC                     ;CC=0101
4530  014262  000252                  +CLN!CLV
4531  014264  005500                  ADC     R0              ;CC=1010
4532  014266  101402                  BLOS    ADC1
4533  014270  102001                  BVC     ADC1
4534  014272  100401                  BMI     ADC2
4535  014274                  ADC1:
4536  014274  104000                  EMT                     ;ADC DID NOT SET CC'S CORRECTLY
4537  014276  052700  077777  ADC2:   BIS     #77777,R0
4538  014302  000277                  SCC                     ;CC=1011
4539  014304  000244                  CLZ
4540  014306  005500                  ADC     R0              ;CC=0101   R0=0
4541  014310  101002                  BHI     ADC3
4542  014312  102401                  BVS     ADC3
4543  014314  100001                  BPL     ADC4
4544  014316                  ADC3:
4545  014316  104000                  EMT                     ;ADC DID NOT SET CC'S CORRECTLY
4546  014320  000277          ADC4:   SCC
4547  014322  000245                  +CLZ!CLC                ;CC=1010
4548  014324  005500                  ADC     R0              ;CC=0100
4549  014326  102403                  BVS     ADC5
4550  014330  103402                  BCS     ADC5
4551  014332  100401                  BMI     ADC5
4552  014334  001401                  BEQ     TS222
4553  014336                  ADC5:
4554  014336  104000                  EMT                     ;ADC DID NOT SET CC'S CORRECTLY
4555
4556          ;*****************************************************************************************
4557          ;
4558          ;       THESE NEXT THREE TESTS VERIFY THE FUNCTIONING OF THE NEG,
4559          ;CMP, AND COM INSTRUCTIONS.  EACH OF THESE INSTRUCTIONS GENERATE
4560          ;THE C AND V BITS IDENTICALLY.  THE CONDITION CODES ARE PRESET,
4561          ;THE INSTRUCTIONS EXECUTED, AND THE RESULTS CHECKED WITH A SERIES
4562          ;OF CONDITIONAL BRANCH INSTRUCTIONS.  THIS PROCEDURE IS REPEATED
4563          ;SEVERAL TIMES WITH DIFFERENT DATA IN ORDER TO GENERATE DIFFERENT
4564          ;COMBINATIONS OF THE C AND V BITS.
4565          ;
4566          ;*****************************************************************************************
4567          ;TEST 222       TEST NEG INSTRUCTION
4568          ;*****************************************************************************************
4569  014340  TS222:
4570  014340  012700  000001          MOV     #1,R0
4571  014344  000277                  SCC                     ;CC=0110
```

```
4572  014346  000251
4573  014350  005400                +CLN!CLC
4574  014352  103003                NEG     R0              ;CC=1001   R0=177777
4575  014354  102402                BCC     NEG1
4576  014356  001401                BVS     NEG1
4577  014360  100401                BEQ     NEG1
4578  014362                        BMI     NEG2
4579  014362  104000        NEG1:   EMT                     ;NEG DID NOT SET CC'S CORRECTLY
4580  014364  042700  077777  NEG2: BIC     #77777,R0
4581  014370  000257                CCC                     ;CC=0100
4582  014372  000264                SEZ
4583  014374  005400                NEG     R0              ;CC=1011   R0=100000
4584  014376  102003                BVC     NEG3
4585  014400  103002                BCC     NEG3
4586  014402  001401                BEQ     NEG3
4587  014404  100401                BMI     NEG4
4588  014406                NEG3:
4589  014406  104000                EMT                     ;NEG DID NOT SET CC'S CORRECTLY
4590  014410  005000        NEG4:   CLR     R0
4591  014412  000277                SCC                     ;CC=1011
4592  014414  000244                CLZ
4593  014416  005400                NEG     R0              ;CC=0100   R0=0
4594  014420  102403                BVS     NEG5
4595  014422  103402                BCS     NEG5
4596  014424  001001                BNE     NEG5
4597  014426  100001                BPL     TS223
4598  014430                NEG5:
4599  014430  104000                EMT                     ;NEG DID NOT SET CC'S CORRECTLY
4600
4601                    ;*********************************************************************************************
4602                    ;TEST 223       TEST CMP INSTRUCTION
4603                    ;*********************************************************************************************
4604  014432            TS223:
4605  014432  012700  000005        MOV     #5,R0
4606  014436  000257                CCC                     ;CC=1010
4607  014440  000271                +SEN!SEC
4608  014442  022700  000005        CMP     #5,R0           ;CC=0101
4609  014446  101002                BHI     CMP1
4610  014450  102401                BVS     CMP1
4611  014452  100001                BPL     CMP2
4612  014454            CMP1:
4613  014454  104000                EMT                     ;CMP DID NOT SET CC'S CORRECTLY
4614  014456  012700  100000  CMP2: MOV     #100000,R0
4615  014462  000277                SCC                     ;CC=1101
4616  014464  000242                CLV
4617  014466  020027  077777        CMP     R0,#77777       ;CC=0010
4618  014472  101402                BLOS    CMP3
4619  014474  102001                BVC     CMP3
4620  014476  100001                BPL     CMP4
4621  014500            CMP3:
4622  014500  104000                EMT                     ;CMP DID NOT SET CC'S CORRECTLY
4623  014502  052700  040000  CMP4: BIS     #40000,R0       ;R0=140000
4624  014506  000257                CCC                     ;CC=0100
4625  014510  000264                SEZ
4626  014512  022700  040000        CMP     #40000,R0       ;CC=1011
4627  014516  102003                BVC     CMP5
```

```
4628  014520  103002                BCC     CMP5
4629  014522  001401                BEQ     CMP5
4630  014524  100401                BMI     CMP6
4631  014526                    CMP5:
4632  014526  104000                EMT                     ;CMP DID NOT SET CC'S CORRECTLY
4633  014530  042700  040000    CMP6: BIC     #40000,R0
4634  014534  000277                SCC                     ;CC=1111
4635  014536  022700  177777        CMP     #-1,R0          ;CC=0000
4636  014542  101402                BLOS    CMP7
4637  014544  102401                BVS     CMP7
4638  014546  100001                BPL     TS224
4639  014550                    CMP7:
4640  014550  104000                EMT                     ;CMP DID NOT SET CC'S CORRECTLY
4641
4642                        ;*****************************************************************************
4643                        ;TEST 224       TEST COM INSTRUCTION
4644                        ;*****************************************************************************
4645  014552                TS224:
4646  014552  012700  177777        MOV     #-1,R0
4647  014556  000257                CCC                     ;CC=1010
4648  014560  000265                +SEC!SEZ
4649  014562  005100                COM     R0              ;CC=0101
4650  014564  101002                BHI     COM1
4651  014566  102401                BVS     COM1
4652  014570  100001                BPL     TS225
4653  014572                    COM1:
4654  014572  104000                EMT                     ;COM DID NOT SET CC'S CORRECTLY
4655
4656                        ;*****************************************************************************
4657                        ;
4658                        ;       THESE NEXT TWO TESTS VERIFY THE FUNCTIONING OF THE SUB
4659                        ;AND SBC INSTRUCTIONS.  BOTH OF THESE INSTRUCTIONS HANDLE THE
4660                        ;C AND V BITS IDENTICALLY.  THE PROCEDURE IS TO PRESET THE CONDITION
4661                        ;CODES, EXECUTE THE INSTRUCTION WITH A PARTICULAR SET OF DATA, AND
4662                        ;THEN CHECK THE RESULTS BY EXECUTING A SERIES OF CONDITIONAL
4663                        ;BRANCHES.  THIS PROCEDURE IS REPEATED SEVERAL TIMES WITH DIFFERENT
4664                        ;DATA PATTERNS TO PROVIDE EVERY COMBINATION OF THE C AND V BITS.
4665                        ;
4666                        ;*****************************************************************************
4667                        ;TEST 225       TEST SUB INSTRUCTION
4668                        ;*****************************************************************************
4669
4670  014574                TS225:
4671  014574  012700  125252        MOV     #125252,R0
4672  014600  000257                CCC                     ;CC=1010
4673  014602  000271                +SEN!SEC
4674  014604  162700  125252        SUB     #125252,R0      ;CC=0101   R0=0
4675  014610  101002                BHI     SUB1
4676  014612  102401                BVS     SUB1
4677  014614  100001                BPL     SUB2
4678  014616                    SUB1:
4679  014616  104000                EMT                     ;SUB DID NOT SET CC'S CORRECTLY
4680  014620  052700  100000    SUB2: BIS     #100000,R0
4681  014624  000277                SCC                     ;CC=1101
4682  014626  000242                CLV
4683  014630  162700  077777        SUB     #77777,R0       ;CC=0010   R0=1
```

```
4684  014634  101402                 BLOS    SUB3
4685  014636  102001                 BVC     SUB3
4686  014640  100001                 BPL     SUB4
4687  014642                 SUB3:
4688  014642  104000                 EMT                     ;
4689  014644  005100         SUB4:   COM     R0              ;R0=177777
4690  014646  000277                 SCC                     ;CC=11111
4691
4692  014650  162700  100000         SUB     #100000,R0      ;CC=0000   R0=77777
4693  014654  101402                 BLOS    SUB5
4694  014656  102401                 BVS     SUB5
4695  014660  100001                 BPL     SUB6
4696  014662                 SUB5:
4697  014662  104000                 EMT                     ;SUB DID NOT SET CC'S CORRECTLY
4698  014664  000257         SUB6:   CCC                     ;CC=0100
4699  014666  000264                 SEZ
4700  014670  162700  140000         SUB     #140000,R0      ;CC=1011
4701  014674  102003                 BVC     SUB7
4702  014676  103002                 BCC     SUB7
4703  014700  001401                 BEQ     SUB7
4704  014702  100401                 BMI     TS226
4705  014704                 SUB7:
4706  014704  104000                 EMT                     ;
4707
4708                         ;*****************************************************************************************
4709                         ;TEST 226       TEST SBC INSTRUCTION
4710                         ;*****************************************************************************************
4711  014706                 TS226:
4712  014706  012700  000001         MOV     #1,R0
4713  014712  000277                 SCC                     ;CC=1011
4714  014714  000244                 CLZ
4715  014716  005600                 SBC     R0              ;CC=0100   R=0
4716  014720  103403                 BCS     SBC1
4717  014722  102402                 BVS     SBC1
4718  014724  100401                 BMI     SBC1
4719  014726  001401                 BEQ     SBC2
4720  014730                 SBC1:
4721  014730  104000                 EMT                     ;SBC DID NOT SET CC'S CORRECTLY
4722  014732  000277         SBC2:   SCC                     ;CC=1010
4723  014734  000245                 +CLZ!CLC
4724  014736  005600                 SBC     R0              ;CC=0100   R=0
4725  014740  103403                 BCS     SBC3
4726  014742  102402                 BVS     SBC3
4727  014744  100401                 BMI     SBC3
4728  014746  001401                 BEQ     SBC4
4729  014750                 SBC3:
4730  014750  104000                 EMT                     ;SBC DID NOT SET CC'S CORRECTLY
4731  014752  000277         SBC4:   SCC                     ;CC=0111
4732  014754  000250                 CLN
4733  014756  005600                 SBC     R0              ;CC=1001   R0=177777
4734  014760  103003                 BCC     SBC5
4735  014762  102402                 BVS     SBC5
4736  014764  001401                 BEQ     SBC5
4737  014766  100401                 BMI     SBC6
4738  014770                 SBC5:
4739  014770  104000                 EMT                     ;SBC DID NOT SET CC'S CORRECTLY
```

```
 4740  014772  042700  077777     SBC6:   BIC     #77777,R0       ;R0=100000
 4741  014776  000277                     SCC                     ;CC=1101
 4742  015000  000242                     CLV
 4743  015002  005600                     SBC     R0              ;CC=0010
 4744  015004  101402                     BLOS    SBC7
 4745  015006  102001                     BVC     SBC7
 4746  015010  100001                     BPL     TS227
 4747  015012                     SBC7:
 4748  015012  104000                     EMT                     ;SBC DID NOT SET CC'S CORRECTLY
 4749
 4750                             ;*****************************************************************************
 4751                             ;
 4752                             ;       THESE NEXT FOUR TESTS VERIFY THE FUNCTIONING OF THE ROL,
 4753                             ;ROR, ASL AND ASR INSTRUCTIONS.  SPECIAL DATA PATTERNS ARE LOADED
 4754                             ;AND ROTATED SEVERAL TIMES FOR EACH TEST.  THE CONDITION CODES
 4755                             ;ARE PRESET BEFORE EACH ROTATION AND THE CONDITION CODES ARE
 4756                             ;CHECKED AFTER EACH ROTATION.  THE FINAL CHECK IN EACH TEST IS
 4757                             ;TO VERIFY THE COMMULATIVE DATA RESULT.  THE DATA PATTERNS HAVE
 4758                             ;BEEN SELECTED TO PRODUCE ALL COMBINATIONS OF THE C AND V BITS.
 4759                             ;
 4760                             ;*****************************************************************************
 4761                             ;TEST 227       TEST ROL INSTRUCTION
 4762                             ;*****************************************************************************
 4763  015014                     TS227:
 4764  015014  012700  144000             MOV     #144000,R0      ;R0=144000
 4765  015020  000257                     CCC                     ;CC=0110
 4766  015022  000266                     +SEZ!SEV
 4767  015024  006100                     ROL     R0              ;CC=1001   R0=110000
 4768  015026  103003                     BCC     ROL1
 4769  015030  102402                     BVS     ROL1
 4770  015032  001401                     BEQ     ROL1
 4771  015034  100401                     BMI     ROL2
 4772  015036                     ROL1:
 4773  015036  104000                     EMT                     ;
 4774  015040  000277             ROL2:   SCC                     ;CC=1100
 4775  015042  000243                     +CLV!CLC
 4776  015044  006100                     ROL     R0              ;CC=0011   R0=020000
 4777  015046  103003                     BCC     ROL3
 4778  015050  102002                     BVC     ROL3
 4779  015052  001401                     BEQ     ROL3
 4780  015054  100001                     BPL     ROL4
 4781  015056                     ROL3:
 4782  015056  104000                     EMT                     ;ROL DID NOT SET CC'S CORRECTLY
 4783  015060  000277             ROL4:   SCC                     ;CC=0111
 4784  015062  000250                     CLN
 4785  015064  006100                     ROL     R0              ;CC=0000   R0=040001
 4786  015066  101402                     BLOS    ROL5
 4787  015070  102401                     BVS     ROL5
 4788  015072  100001                     BPL     ROL6
 4789  015074                     ROL5:
 4790  015074  104000                     EMT                     ;ROL DID NOT SET CC'S CORRECTLY
 4791  015076  000257             ROL6:   CCC                     ;CC=0101
 4792  015100  000265                     +SEZ!SEC
 4793  015102  006100                     ROL     R0              ;CC=1010   R0=100003
 4794  015104  101405                     BLOS    ROL7
 4795  015106  102004                     BVC     ROL7
```

```
4796  015110  100003
4797  015112  022700  100003          BPL     ROL7
4798  015116  001401                  CMP     #100003,R0
4799  015120                          BEQ     TS230
4800  015120  104000          ROL7:
4801                                  EMT                     ;ROL MALFUNCTIONED
4802          ;*************************************************************************************
4803          ;TEST 230       TEST ROR INSTRUCTION
4804  015122  ;*************************************************************************************
4805  015122  012700  000023  TS230:
4806  015126  000277                  MOV     #23,R0          ;R0=23
4807  015130  000250                  SCC                     ;CC=0111
4808  015132  006000                  CLN
4809  015134  102403                  ROR     R0              ;CC=1001   R0=100011
4810  015136  103002                  BVS     ROR1
4811  015140  001401                  BCC     ROR1
4812  015142  100401                  BEQ     ROR1
4813  015144                          BMI     ROR2
4814  015144  104000          ROR1:
4815  015146  000257                  EMT                     ;ROR DID NOT SET CC'S CORRECTLY
4816  015150  000274          ROR2:   CCC                     ;CC=1100
4817  015152  006000                  +SEN!SEZ
4818  015154  102003                  ROR     R0              ;CC=0011   R0=040004
4819  015156  103002                  BVC     ROR3
4820  015160  001401                  BCC     ROR3
4821  015162  100001                  BEQ     ROR3
4822  015164                          BPL     ROR4
4823  015164  104000          ROR3:
4824  015166  000277                  EMT                     ;ROR DID NOT SET CC'S CORRECTLY
4825  015170  000241          ROR4:   SCC                     ;CC=1110
4826  015172  006000                  CLC
4827  015174  101403                  ROR     R0              ;CC=0000   R0=020002
4828  015176  102402                  BLOS    ROR5
4829  015200  001401                  BVS     ROR5
4830  015202  100001                  BEQ     ROR5
4831  015204                          BPL     ROR6
4832  015204  104000          ROR5:
4833  015206  000257                  EMT                     ;ROR DID NOT SET CC'S CORRECTLY
4834  015210  000265          ROR6:   CCC                     ;CC=0101
4835  015212  006000                  +SEC!SEZ
4836  015214  101402                  ROR     R0              ;CC=1010   R0=110001
4837  015216  102001                  BLOS    ROR7
4838  015220  100401                  BVC     ROR7
4839  015222                          BMI     TS231
4840  015222  104000          ROR7:
4841                                  EMT                     ;ROR DID NOT PRODUCE CORRECT RESULTS
4842          ;*************************************************************************************
4843          ;TEST 231       TEST ASL INSTRUCTION
4844  015224  ;*************************************************************************************
4845  015224  012700  144000  TS231:
4846  015230  000257                  MOV     #144000,R0      ;R0=14000
4847  015232  000271                  CCC                     ;CC=0110
4848  015234  006300                  +SEN!SEC
4849  015236  103003                  ASL     R0              ;CC=1001   R0=110000
4850  015240  102402                  BCC     ASL1
4851  015242  001401                  BVS     ASL1
                                      BEQ     ASL1
```

```
4852  015244  100401                  BMI     ASL2
4853  015246                  ASL1:
4854  015246  104000                  EMT                     ;
4855  015250  000277          ASL2:   SCC                     ;CC=1100
4856  015252  000243                  +CLV!CLC
4857  015254  006300                  ASL     R0              ;CC=0011   R0=020000
4858  015256  103003                  BCC     ASL3
4859  015260  102002                  BVC     ASL3
4860  015262  001401                  BEQ     ASL3
4861  015264  100001                  BPL     ASL4
4862  015266                  ASL3:
4863  015266  104000                  EMT                     ;ASL DID NOT SET CC'S CORRECTLY
4864  015270  000277          ASL4:   SCC                     ;CC=0111
4865  015272  000250                  CLN
4866  015274  006300                  ASL     R0              ;CC=0000   R0=040000
4867  015276  101402                  BLOS    ASL5
4868  015300  102401                  BVS     ASL5
4869  015302  100001                  BPL     ASL6
4870  015304                  ASL5:
4871  015304  104000                  EMT                     ;ASL DID NOT SET CC'S CORRECTLY
4872  015306  000257          ASL6:   CCC                     ;CC=0101
4873  015310  000265                  +SEZ!SEC
4874  015312  006300                  ASL     R0              ;CC=1010   R0=100000
4875  015314  103406                  BCS     ASL7
4876  015316  001405                  BEQ     ASL7
4877  015320  102004                  BVC     ASL7
4878  015322  100003                  BPL     ASL7
4879  015324  022700  100000          CMP     #100000,R0
4880  015330  001401                  BEQ     TS232
4881  015332                  ASL7:
4882  015332  104000                  EMT                     ;ASL MALFUNCTIONED
4883                          ;*******************************************************************************
4884                          ;TEST 232       TEST ASR INSTRUCTION
4885                          ;*******************************************************************************
4886  015334                  TS232:
4887  015334  012700  100023          MOV     #100023,R0      ;R0=100023
4888  015340  000277                  SCC                     ;CC=0110
4889  015342  000250                  CLN
4890  015344  006200                  ASR     R0              ;CC=1001   RP=140011
4891  015346  102403                  BVS     ASR1
4892  015350  103002                  BCC     ASR1
4893  015352  001401                  BEQ     ASR1
4894  015354  100401                  BMI     ASR2
4895  015356                  ASR1:
4896  015356  104000                  EMT                     ;ASR DID NOT SET CC'S CORRECTLY
4897  015360  042700  100000  ASR2:   BIC     #100000,R0      ;R0=40011
4898  015364  000277                  SCC                     ;CC=1100
4899  015366  000243                  +CLV!CLC
4900  015370  006200                  ASR     R0              ;CC=0011   R0=020004
4901  015372  102003                  BVC     ASR3
4902  015374  103002                  BCC     ASR3
4903  015376  001401                  BEQ     ASR3
4904  015400  100001                  BPL     ASR4
4905  015402                  ASR3:
4906  015402  104000                  EMT                     ;ASR DID NOT SET CC'S CORRECTLY
4907  015404  000277          ASR4:   SCC                     ;CC=1111
```

```
                                        ASR     R0              ;CC=0000   R0=010002
                                        BLOS    ASR5
                                        BVS     ASR5
                                        BEQ     ASR5
                                        BPL     ASR6
                                ASR5:
                                        EMT                     ;ASR DID NOT SET CC'S CORRECTLY
                                ASR6:   BIS     #100000,R0      ;R0=110002
                                        CCC                     ;CC=0101
                                        +SEZ!SEC
                                        ASR     R0              ;C=1010   R0=144001
                                        BLOS    ASR7
                                        BVC     ASR7
                                        BPL     ASR7
                                        BEQ     ASR7
                                        CMP     #144001,R0      ;CHECK RESULT OF ASR'S
                                        BEQ     TS233
                                ASR7:
                                        EMT                     ;ASR DID NOT FUNCTION CORRECTLY


                                ;*******************************************************************************
                                ;TEST 233        TEST RORB INSTRUCTION
                                ;*******************************************************************************
                                TS233:
                                        MOVB    #4,R1   ;LOAD REGISTER
                                        CCC                     ;CLEAR ALL FLAGS
                                        RORB    R1              ;SHIFT  BYTE RIGHT
                                        RORB    R1              ;SHIFT BYTE RIGHT
                                        CMPB    #1,R1           ;CHECK RESULT
                                        BEQ     RORB1
                                        EMT                     ;RORB DID NOT FUNCTION CORRECTLY
                                RORB1:  RORB    R1              ;SHIFT BYTE RIGHT
                                        BMI     RORB2           ;CC=7?
                                        BNE     RORB2
                                        BVC     RORB2
                                        BCS     RORB3
                                RORB2:
                                        EMT                     ;RORB DID NOT SET CC'S CORRECTLY
                                RORB3:  RORB    R1              ;SHIFT BYTE RIGHT
                                        BPL     RORB4           ;CC=12
                                        BLOS    RORB4
                                        BVS     RORB5
                                RORB4:
                                        EMT                     ;RORB DID NOT SET CC CORRECTLY
                                RORB5:  CMPB    #200,R1         ;CHECK RESULT
                                        BEQ     RORB7
                                        EMT                     ;RORB DID NOT FUNCTION CORRECTLY
                                RORB7:
                                ;ROTATE ODD BYTE
                                        CLR     R0              ;MAKE R0 ZERO
                                        MOV     #025125,(R0)    ;PUT STARTING VALUE IN LOC. 0
                                        INC     R0              ;MAKE R0 POINT TO ODD BYTE
                                        CCC                     ;CLEAR ALL CC
                                        SEC                     ;SEC CARRY BIT
```

```
4964  015550  106010                  RORB    (R0)            ;SHIFT BYTE RIGHT
4965  015552  100002                  BPL     RORB10          ;CC=12?
4966  015554  101401                  BLOS    RORB10
4967  015556  102401                  BVS     RORB11
4968  015560                  RORB10:
4969  015560  104000                  EMT                     ;RORB DID NOT SET CC'S CORRECTLY
4970  015562  022737  112525  000000  RORB11: CMP     #112525,@#0     ;CHECK RESULT
4971  015570  001401                  BEQ     RORB12
4972  015572  104000                  EMT                     ;RORB DID NOT FUNCTION CORRECTLY
4973  015574  106010          RORB12: RORB    (R0)            ;SHIFT BYTE RIGHT
4974  015576  100403                  BMI     RORB13          ;CC=3?
4975  015600  001402                  BEQ     RORB13
4976  015602  102001                  BVC     RORB13
4977  015604  103401                  BCS     RORB14
4978  015606                  RORB13:
4979  015606  104000                  EMT                     ;RORB DID NOT SET CC CORRECTLY
4980  015610  022737  045125  000000  RORB14: CMP     #045125,@#0     ;CHECK RESULT
4981  015616  001401                  BEQ     TS234
4982  015620  104000                  EMT                     ;RORB DID NOT FUNCTION CORRECTLY
4983
4984
4985                          ;*****************************************************************************
4986                          ;TEST 234       TEST ASLB INSTRUCTION
4987                          ;*****************************************************************************
4988  015622                  TS234:
4989  015622  112701  000040          MOVB    #40,R1          ;LOAD REGISTER
4990  015626  000257                  CCC                     ;CLEAR ALL CONDITION CODES
4991  015630  106301                  ASLB    R1              ;SHIFT BYTE LEFT
4992  015632  106301                  ASLB    R1              ;SHIFT BYTE LEFT
4993  015634  100002                  BPL     ASLB2           ;CHECK CC=12
4994  015636  101401                  BLOS    ASLB2
4995  015640  102401                  BVS     ASLB3
4996  015642                  ASLB2:
4997  015642  104000                  EMT                     ;ASLB DID NOT SET CONDITION CODE CORRECTLY
4998  015644  022701  000200  ASLB3:  CMP     #200,R1         ;CHECK RESULT
4999  015650  001401                  BEQ     ASLB1
5000  015652  104000                  EMT                     ;ASLB DID NOT FUNCTION CORRECTLY
5001  015654  106301          ASLB1:  ASLB    R1              ;SHIFT BYTE LEFT
5002  015656  100403                  BMI     ASLB4           ;CHECK CC=7?
5003  015660  001002                  BNE     ASLB4
5004  015662  102001                  BVC     ASLB4
5005  015664  103401                  BCS     TS235
5006  015666                  ASLB4:
5007  015666  104000                  EMT                     ;ASLB   DID NOT SET CC'S CORRECTLY
5008
5009
5010                          ;*****************************************************************************
5011                          ;TEST 235       TEST ASRB INSTRUCTION
5012                          ;*****************************************************************************
5013  015670                  TS235:
5014  015670  112701  000004          MOVB    #4,R1           ;SET UP STARTING DATA
5015  015674  000257                  CCC                     ;CLEAR ALL CONDITION CODES
5016  015676  106201                  ASRB    R1              ;SKIFT BYTE RIGHT
5017  015700  106201                  ASRB    R1              ;SHIFT BYTE RIGHT
5018  015702  122701  000001          CMPB    #1,R1           ;CHECK DATA
5019  015706  001401                  BEQ     ASRB1
```

```
5020  015710  104000                  EMT                     ;ASRB DID NOT SHIFT DATA CORRECTLY
5021  015712  106201          ASRB1:  ASRB    R1              ;SHIFT BYTE RIGHT
5022  015714  100403                  BMI     ASRB2           ;CHECK CONDITION CODE = 7?
5023  015716  001002                  BNE     ASRB2
5024  015720  102001                  BVC     ASRB2
5025  015722  103401                  BCS     ASRB3
5026  015724                  ASRB2:
5027  015724  104000                  EMT                     ;ASRB DID NOT SET CC'S CORRECTLY
5028  015726  106201          ASRB3:  ASRB    R1              ;SHIFT BYTE RIGHT
5029  015730  103401                  BCS     ASRB4           ;CHECK CC=4
5030  015732  001401                  BEQ     ASRB5
5031  015734                  ASRB4:
5032  015734  104000                  EMT                     ;ASRB DID NOT SET CC'S CORRECTLY
5033  015736  112701  000202  ASRB5:  MOVB    #202,R1         ;PUT STARTING DATA IN REGISTER
5034  015742  106201                  ASRB    R1              ;SHIFT BYTE RIGHT
5035  015744  106201                  ASRB    R1              ;SHIFT BYTE RIGHT
5036  015746  100003                  BPL     ASRB6           ;CHECK CC'S =11?
5037  015750  001402                  BEQ     ASRB6
5038  015752  102401                  BVS     ASRB6
5039  015754  103401                  BCS     ASRB7
5040  015756                  ASRB6:
5041  015756  104000                  EMT                     ;ASRB DID NOT SET CC'S CORRECTLY
5042  015760  122701  000340  ASRB7:  CMPB    #340,R1         ;CHECK RESULT
5043  015764  001401                  BEQ     TS236
5044  015766  104000                  EMT                     ;ASRB DID NOT SHIFT DATA CORRECTLY
5045
5046                          ;*******************************************************************************
5047                          ;
5048                          ;       THIS TEST VERIFIES THE SXT INSTRUCTION. CONDITION  CODES
5049                          ;ARE PRESET IN EACH OF THE TWO POSSIBLE CASES. WITH THE N-BIT SET,
5050                          ;THE TEST CHECKS FOR ALL ONES IN THE DESTINATION. WITH THE N-BIT
5051                          ;CLEAR, THE DESTINATION SHOULD CONTAIN ALL ZEROES. THE DATA
5052                          ;IS VERIFIED BY CONDITIONAL BRANCHES.
5053                          ;
5054                          ;*******************************************************************************
5055                          ;TEST 236       TEST THE SXT INSTRUCTION
5056                          ;*******************************************************************************
5057  015770                  TS236:
5058  015770  005000                  CLR     R0
5059  015772  000277                  SCC                     ;SET CC=1011
5060  015774  000244                  CLZ
5061  015776  006700                  SXT     R0              ;TRY SXT
5062  016000  100006                  BPL     SXT0            ;TEST CC=1001
5063  016002  001405                  BEQ     SXT0
5064  016004  102404                  BVS     SXT0
5065  016006  103003                  BCC     SXT0
5066  016010  022700  177777          CMP     #-1,R0          ;CHECK DATA RESULT
5067  016014  001401                  BEQ     SXT1
5068  016016                  SXT0:
5069  016016  104000                  EMT                     ;RESULTS OF SXT INCORRECT
5070  016020  005000          SXT1:   CLR     R0              ;R0=0
5071  016022  005010                  CLR     (R0)            ;LOC. 0=0
5072  016024  005110                  COM     (R0)            ;LOC. 0=177777
5073  016026  000257                  CCC                     ;SET CC=0110
5074  016030  000266                  *SEZ!SEV
5075  016032  006710                  SXT     (R0)
```

```
5076  016034  001005          BNE     SXT2            ;TEST CC=0100
5077  016036  103404          BCS     SXT2
5078  016040  102403          BVS     SXT2
5079  016042  100402          BMI     SXT2
5080  016044  005710          TST     (R0)
5081  016046  001401          BEQ     TS237
5082  016050                  SXT2:
5083  016050  104000          EMT                     ;RESULTS OF SXT INCORRECT
5084                  ;*****************************************************************************
5085                  ;
```

```
                                ;       THIS TEST VERIFIES THE XOR INSTRUCTION. UNIQUE PATTERNS
                                ;OF ONES AND ZEROES ARE MOVED TO DATA REGISTERS R0 AND R1.
                                ;AFTER THE FIRST XOR INSTRUCTION R0=36146. AN XOR IS THEN
                                ;EXECUTED WITH THIS NEW VALUE AND THE CONTENTS OF R1 TO
                                ;REPRODUCE THE ORIGINAL VALUE IF R0=31525.
                                ;
                                ;*****************************************************************************************
                                ;TEST 237       TEST THE XOR INSTRUCTION
                                ;*****************************************************************************************
016052                          TS237:
016052  012700  007463                MOV     #7463,R0        ;SET UP R0
016056  012701  031525                MOV     #31525,R1       ;SET UP R1
016062  000277                        SCC                     ;SET CC=1110
016064  000241                        CLC
016066  074100                        XOR     R1,R0           ;TRY XOR
016070  101406                        BLOS    XOR1            ;CC=0000?
016072  102405                        BVS     XOR1
016074  001404                        BEQ     XOR1
016076  100403                        BMI     XOR1
016100  022700  036146                CMP     #36146,R0       ;DATA RESULT CORRECT?
016104  001401                        BEQ     XOR2
016106                          XOR1:
016106  104000                        EMT                     ;
016110  010104                  XOR2: MOV     R1,R4
016112  000261                        SEC                     ;CC=1110
016114  000241                        CLC
016116  074400                        XOR     R4,R0           ;TRY XOR MODE 0,0
016120  101406                        BLOS    XOR3            ;CC=0000?
016122  102405                        BVS     XOR3
016124  001404                        BEQ     XOR3
016126  100403                        BMI     XOR3
016130  022700  007463                CMP     #7463,R0
016134  001401                        BEQ     TS240
016136                          XOR3:
016136  104000                        EMT                     ;RESULT OF XOR INCORRECT
                                ;*****************************************************************************************
                                ;
                                ;       THIS TEST VERIFIES THE SOB INSTRUCTION. R4 IS USED AS A
                                ;COUNTER WHILE R0 IS THE ADDRESS REGISTER. CONDITIONAL
                                ;BRANCHES ARE USED TO VERIFY PROPER TRANSFER OF CONTROL
                                ;WHILE R4 IS CHECKED TO INSURE PROPER DECREMENTING OF R0.
                                ;
                                ;*****************************************************************************************
                                ;TEST 240       TEST SOB INSTRUCTION
                                ;*****************************************************************************************
016140                          TS240:
016140  012700  000525                MOV     #525,R0
016144  010004                        MOV     R0,R4
016146  000277                        SCC                     ;SET CC=1111
016150  101002                  SOB1: BHI     SOB2            ;CC=1111?
016152  100001                        BPL     SOB2
016154  102401                        BVS     SOB3
016156                          SOB2:
016156  104000                        EMT                     ;
016160  005304                  SOB3: DEC     R4              ;COUNT ITERATIONS
016162  000277                        SCC                     ;CC=1111
```

```
                        SOB     R0,SOB1         ;DO SOB W/ R0
                        BHI     SOB4            ;CHECK CC=1111
                        BPL     SOB4
                        BVC     SOB4
                        TST     R4              ;ITERATION COUNT OK?
                        BEQ     TS241
                SOB4:
                        EMT                     ;INCORRECT # OF BRANCHES OR CC'S CHANGED
                ;***************************************************************************
                ;
                ;       THIS TEST VERIFIES THE MARK INSTRUCTION. THE EFFECTS
                ;OF THE MARK INSTRUCTION ARE SIMULATED BY THE PROGRAM INSTRUCTIONS.
                ;THE CONTENTS OF R5 AND THE STACK POINTER ARE CHECKED AFTER EACH
                ;OF THE TWO ROUTINES IN THE TEST.
                ;
                ;***************************************************************************
                ;TEST 241       TEST MARK INSTRUCTION
                ;***************************************************************************
                TS241:
                        MOV     #STBOT,SP
                        MOV     #125252,-(SP)   ;PUT R5 VALUE ON STACK
                        SUB     #74,SP          ;EFFECTIVELY PUT 36 ARGUMENTS ON STACK
                        MOV     #MRK1,R5        ;SET NEW PC IN R5
                        MOV     #6436,-(SP)     ;PUT MARK 36 INST. ON STACK
                        SCC                     ;SET CC=1111
                        JMP     (SP)            ;XFER CONTL TO MARK 36 INST. ON STACK
                        EMT                     ;MARK INST. SHOULD HAVE JUMPED TO MRK1
                MRK1:   BHI     MRK2            ;TEST CC UNAFFECTED
                        BPL     MRK2            ;IE. CC=1111
                        BVC     MRK2
                        CMP     R5,#125252      ;CHECK R5 RESTORED FROM STACK
                        BNE     MRK2
                        CMP     #STBOT,R6       ;CHECK STACK POINTER READJUSTED CORRECTLY.
                        BEQ     MRK3
                MRK2:
                        EMT                     ;RESULTS OF MARK INCORRECT
                MRK3:   MOV     #52525,-(SP)
                        MOV     #6400,-(SP)     ;PUT MARK 0 INST. ON STACK
                        MOV     SP,R5           ;SET ADDR. OF MARK INST. IN R5
                        JSR     PC,@#MRK4       ;DO JSR
                        JMP     @#MRK5          ;
                MRK4:   RTS     R5              ;DO RTS WITH R5 TO MARK INST ON STACK
                        EMT                     ;RTS,MARK SEQUENCE FAILED
                MRK5:   CMP     #STBOT,R6       ;STACK ADJUSTED CORRECTLY
                        BNE     MRK6            ;IF NOT:  BR
                        CMP     #52525,R5       ;CHECK IF R5 RESTORED FROM STACK
                        BEQ     TS242
                MRK6:
                        EMT                     ;RESULTS OF MARK INCORRECT
                        PS=177776
                ;***************************************************************************
                ;
                ;       THESE NEXT SEVEN TESTS VERIFY THE MTPS INSTRUCTION IN ALL
                ;       MODES. THE PSW IS DEFINED BY AN EQUATE STATEMENT BEFORE THE
                ;       FIRST MTPS TEST. IN EACH TEST A PATTERN OF ONES AND
                ;       ZEROES IS SET IN A DATA REGISTER AND MOVED TO THE PSW.
```

```
                        ;       THE DATA IN THE PSW, AND THE DATA REGISTER ADDRESS,
                        ;       ARE CHECKED TO VERIFY PROPER EXECUTION OF THE INSTRUCTION.
                        ;
                        ;*******************************************************************************
                        ;TEST 242       TEST MTPS INSTRUCTION
                        ;*******************************************************************************
016324                  TS242:
016324  012700  000377          MOV     #377,R0
016330  000257                  CCC
016332  106400                  MTPS    R0
016334  022767  000357  161434  CMP     #357,PS
016342  001401                  BEQ     MTPS1
016344  104000                  EMT                     ;MTPS FAILED
016346  005000          MTPS1:  CLR     R0
016350  005010                  CLR     (R0)
016352  000277                  SCC                     ;CC=1111
016354  106410                  MTPS    (R0)            ;TRY MTPS MODE 1
016356  100403                  BMI     MTPS1A          ;CHECK PS
016360  102402                  BVS     MTPS1A
016362  103401                  BCS     MTPS1A
016364  001001                  BNE     TS243
016366                  MTPS1A:
016366  104000                  EMT                     ;MTPS FAILED
                        ;*******************************************************************************
                        ;TEST 243       TEST MTPS MODE 2
                        ;*******************************************************************************
016370                  TS243:
016370  005000                  CLR     R0              ;R0=0
016372  012710  177777          MOV     #-1,(R0)        ;LOC. 0=-1
016376  005037  177776          CLR     @#PS            ;PS=0
016402  106420                  MTPS    (R0)+           ;TRY MTPS W/MODE 2
016404  022737  000357  177776  CMP     #357,@#PS       ;CHECK DATA
016412  001401                  BEQ     MTPS2
016414  104000                  EMT                     ;DEST. DATA INCORRECT
016416  022700  000001  MTPS2:  CMP     #1,R0           ;CHECK DEST. REGISTER.
016422  001401                  BEQ     TS244
016424  104000                  EMT                     ;DEST REGISTER NOT INCREMENTED BY 1
                        ;*******************************************************************************
                        ;TEST 244       TEST MTPS MODE 3
                        ;*******************************************************************************
016426                  TS244:
016426  012700  000402          MOV     #402,R0         ;R0=402
016432  005010                  CLR     (R0)            ;LOC. 402=0
016434  012737  052652  000000  MOV     #52652,@#0      ;LOC. 0=52652
016442  005037  177776          CLR     @#PS            ;PS=0
016446  106430                  MTPS    @(R0)+          ;TRY MTPS W/MODE 3
016450  022737  000252  177776  CMP     #252,@#PS       ;CHECK DEST. DATA
016456  001401                  BEQ     MTPS3
016460  104000                  EMT                     ;DEST. DATA INCORRECT
016462  022700  000404  MTPS3:  CMP     #404,R0         ;CHECK MODE 3 REGISTER.
016466  001401                  BEQ     TS245
016470  104000                  EMT                     ;MODE 3 REGISTER INCORRECT
                        ;*******************************************************************************
```

```
5254
5255
5256  016472
5257  016472  012700  000001              TS245:
5258  016476  012737  125125  000000              MOV     #1,R0           ;R0=1
5259  016504  005037  177776                    MOV     #125125,@#0     ;LOC. 0 = 125125
5260  016510  106440                             CLR     @#PS            ;PS=0
5261  016512  022737  000105  177776              MTPS    -(R0)           ;TRY MTPS W/MODE 4
5262  016520  001401                             CMP     #105,@#PS       ;CHECK DEST. DATA
5263  016522  104000                             BEQ     MTPS4
5264  016524  005700                             EMT                     ;DEST. DATA INCORRECT
5265  016526  001401                     MTPS4:  TST     R0              ;CHECK MODE 4 REGISTER
5266  016530  104000                             BEQ     TS246
5267                                             EMT                     ;MODE 4 REGISTER NOT DECREMENTED BY 1
5268                                     ;*******************************************************************************
5269                                     ;TEST 246       TEST MTPS MODE 5
5270                                     ;*******************************************************************************
5271  016532                             TS246:
5272  016532  012700  000404                     MOV     #404,R0         ;R0=404
5273  016536  012737  177400  000000             MOV     #177400,@#0     ;LOC. 0=177400
5274  016544  000277                             SCC                     ;SET ALL COND. CODES
5275  016546  106450                             MTPS    @-(R0)          ;TRY MTPS W/MODE 5
5276  016550  005737  177776                    TST     @#PS            ;CHECK DEST. DATA.
5277  016554  001401                             BEQ     MTPS5
5278  016556  104000                             EMT                     ;DESTINATION DATA INCORRECT
5279  016560  022700  000402             MTPS5:  CMP     #402,R0         ;CHECK MODE 5 REGISTER
5280  016564  001401                             BEQ     TS247
5281  016566  104000                             EMT                     ;MODE 5 REGISTER NOT DECREMENTED BY 2
5282
5283                                     ;*******************************************************************************
5284                                     ;TEST 247       TEST MTPS MODE 6
5285                                     ;*******************************************************************************
5286  016570                             TS247:
5287  016570  012737  052652  000000             MOV     #52652,@#0      ;LOC. 0=52652
5288  016576  012700  000406                    MOV     #406,R0         ;R0=406
5289  016602  005037  177776                    CLR     @#PS            ;PS=0
5290  016606  106460  177372                    MTPS    -406(R0)        ;TRY MTPS W/MODE 6
5291  016612  022737  000252  177776             CMP     #252,@#PS       ;CHECK DEST. DATA
5292  016620  001401                             BEQ     MTPS6
5293  016622  104000                             EMT                     ;DEST. DATA INCORRECT
5294  016624  022700  000406             MTPS6:  CMP     #406,R0         ;CHECK MODE 6 REGISTER
5295  016630  001401                             BEQ     TS250
5296  016632  104000                             EMT                     ;MODE 6 REGISTER MODIFIED
5297
5298                                     ;*******************************************************************************
5299                                     ;TEST 250       TEST MTPS MODE 7
5300                                     ;*******************************************************************************
5301  016634                             TS250:
5302  016634  012737  052652  000000             MOV     #52652,@#0      ;LOC. 0=52652
5303  016642  012700  000410                    MOV     #410,R0         ;R0=410
5304  016646  005037  177776                    CLR     @#PS            ;PS=0
5305  016652  106470  177776                    MTPS    @-2(R0)         ;TRY MTPS W/MODE 7
5306  016656  022737  000105  177776             CMP     #105,@#PS       ;CHECK DEST. DATA
5307  016664  001401                             BEQ     MTPS7
5308  016666  104000                             EMT                     ;DESTINATION DATA INCORRECT
5309  016670  022700  000410             MTPS7:  CMP     #410,R0         ;CHECK MODE 7 REGISTER
```

```
5310  016674  001401                        BEQ     TS251
5311  016676  104000                        EMT                     ;MODE 7 REGISTER MODIFIED
5312
5313                                ;*******************************************************************************
5314                                ;
5315                                ;       THESE NEXT SEVEN TESTS VERIFY THE MFPS INSTRUCTION IN ALL
5316                                ;MODES. IN EACH TEST, A PATTERN OF ONES AND ZEROES IS MOVED TO THE
5317                                ;PSW, AND AN MFPS INSTRUCTION MOVES THE DATA TO A LOCATION SETUP
5318                                ;BY R0, EITHER DIRECTLY OR INDIRECTLY. CONDITIONAL BRANCHES ARE
5319                                ;USED TO CHECK PROPER ADDRESSING AND DATA.
5320                                ;
5321                                ;*******************************************************************************
5322                                ;TEST 251       TEST MFPS INSTRUCTION
5323                                ;*******************************************************************************
5324  016700                        TS251:
5325  016700  012737  000377  177776        MOV     #377,@#PS
5326  016706  106700                        MFPS    R0
5327  016710  022700  177757               CMP     #177757,R0
5328  016714  001401                        BEQ     MFPS1
5329  016716  104000                        EMT                     ;MFPS FAILED
5330
5331  016720  005000                MFPS1:  CLR     R0
5332  016722  012737  177777  000000        MOV     #-1,@#0
5333  016730  005037  177776               CLR     @#PS
5334  016734  106710                        MFPS    (R0)
5335  016736  105737  000000               TSTB    @#0
5336  016742  001401                        BEQ     TS252
5337  016744  104000                        EMT                     ;MFPS FAILED
5338
5339                                ;*******************************************************************************
5340                                ;TEST 252       TEST MFPS MODE 2
5341                                ;*******************************************************************************
5342  016746                        TS252:
5343  016746  005000                        CLR     R0              ;R0=0
5344  016750  005010                        CLR     (R0)            ;LOC. 0=0
5345  016752  012737  000377  177776        MOV     #377,@#PS       ;SET PS=357
5346  016760  106720                        MFPS    (R0)+           ;TRY MFPS W/MODE 2
5347  016762  103003                        BCC     MFPS2A          ;BR TO ERROR IF C BIT CLEAR
5348  016764  102402                        BVS     MFPS2A          ;BR TO ERROR IF V BIT SET
5349  016766  001401                        BEQ     MFPS2A          ;BR TO ERROR IF Z BIT SET
5350  016770  100401                        BMI     MFPS2B
5351  016772                        MFPS2A:
5352  016772  104000                        EMT                     ;COND. CODES INCORRECT
5353  016774  022737  000357  000000 MFPS2B: CMP     #357,@#0        ;CHECK DEST. DATA
5354  017002  001401                        BEQ     MFPS2C
5355  017004  104000                        EMT                     ;DEST. DATA INCORRECT
5356  017006  022700  000001        MFPS2C: CMP     #1,R0           ;CHECK MODE 2 REGISTER
5357  017012  001401                        BEQ     TS253
5358  017014  104000                        EMT                     ;MODE 2 REGISTER NOT INCREMENTED 1
5359
5360                                ;*******************************************************************************
5361                                ;TEST 253       TEST MFPS MODE 3
5362                                ;*******************************************************************************
5363  017016                        TS253:
5364  017016  012700  000406               MOV     #406,R0         ;R0=406
5365  017022  005037  000000               CLR     @#0             ;LOC. 0=0
```

```
5366  017026  012737  000252  177776          MOV     #252,@#PS       ;PS=252
5367  017034  106730                          MFPS    @(R0)+          ;TRY MFPS WITH MODE 3
5368  017036  103403                          BCS     MFPS3A          ;BR TO ERROR IF C-BIT SET
5369  017040  102402                          BVS     MFPS3A          ;BR TO ERROR IF V-BIT SET
5370  017042  001401                          BEQ     MFPS3A          ;BR TO ERROR IF Z-BIT SET
5371  017044  100401                          BMI     MFPS3B
5372  017046                          MFPS3A:
5373  017046  104000                          EMT                     ;CONDITION CODES INCORRECT
5374  017050  022737  125000  000000  MFPS3B: CMP     #125000,@#0     ;CHECK DEST. DATA
5375  017056  001401                          BEQ     MFPS3C
5376  017060  104000                          EMT                     ;DEST DATA INCORRECT
5377  017062  020027  000410          MFPS3C: CMP     R0,#410         ;CHECK MODE 3 REGISTER.
5378  017066  001401                          BEQ     TS254
5379  017070  104000                          EMT                     ;MODE 3 REGISTER NOT INCREMENTED BY 2
5380
5381                                  ;*****************************************************************************
5382                                  ;TEST 254       TEST MFPS MODE 4
5383                                  ;*****************************************************************************
5384  017072                          TS254:
5385  017072  012700  000002                  MOV     #2,R0           ;R0=2
5386  017076  005037  000000                  CLR     @#0             ;LOC. 0=0
5387  017102  012737  000125  177776          MOV     #125,@#PS       ;PS=125
5388  017110  106740                          MFPS    -(R0)           ;TRY MFPS W/MODE 4
5389  017112  103003                          BCC     MFPS4A          ;BR TO ERROR IF C-BIT CLEAR
5390  017114  102402                          BVS     MFPS4A          ;BR TO ERROR IF V-BIT SET
5391  017116  001401                          BEQ     MFPS4A          ;BR TO ERROR IF Z-BIT SET
5392  017120  100001                          BPL     MFPS4B
5393  017122                          MFPS4A:
5394  017122  104000                          EMT                     ;COND. CODES INCORRECT
5395  017124  022737  042400  000000  MFPS4B: CMP     #42400,@#0      ;CHECK DEST. DATA
5396  017132  001401                          BEQ     MFPS4C
5397  017134  104000                          EMT                     ;DEST. DATA INCORRECT
5398  017136  020027  000001          MFPS4C: CMP     R0,#1           ;CHECK MODE 4 REGISTER
5399  017142  001401                          BEQ     TS255
5400  017144  104000                          EMT                     ;MODE 4 REGISTER NOT DECREMENTED BY 1
5401
5402                                  ;*****************************************************************************
5403                                  ;TEST 255       TEST MFPS MODE 5
5404                                  ;*****************************************************************************
5405  017146                          TS255:
5406  017146  012700  000410                  MOV     #410,R0         ;R0=410
5407  017152  012737  177777  000000          MOV     #-1,@#0         ;LOC. 0=-1
5408  017160  005037  177776                  CLR     @#PS            ;PS=0
5409  017164  106750                          MFPS    @-(R0)          ;TRY MFPS W/MODE 5
5410  017166  103403                          BCS     MFPS5A          ;BR TO ERROR IF C-BIT SET
5411  017170  102402                          BVS     MFPS5A          ;BR TO ERROR IF V-BIT SET
5412  017172  100401                          BMI     MFPS5A          ;BR TO ERROR IF N-BIT SET
5413  017174  001401                          BEQ     MFPS5B
5414  017176                          MFPS5A:
5415  017176  104000                          EMT                     ;COND. CODES INCORRECT
5416  017200  022737  000377  000000  MFPS5B: CMP     #377,@#0        ;CHECK DEST. DATA
5417  017206  001401                          BEQ     MFPS5C
5418  017210  104000                          EMT                     ;DEST DATA INCORRECT
5419  017212  020027  000406          MFPS5C: CMP     R0,#406         ;CHECK MODE 5 REGISTER
5420  017216  001401                          BEQ     TS256
5421  017220  104000                          EMT                     ;MODE 5 REGISTER NOT DECREMENTED BY 2
```

```
                                ;*****************************************************************************
                                ;TEST 256       TEST MFPS MODE 6
                                ;*****************************************************************************
017222                          TS256:
017222  012700  000401                  MOV     #401,R0         ;R0=410
017226  005037  000000                  CLR     @#0             ;LOC. 0=0
017232  012737  000252  177776          MOV     #252,@#PS       ;PS=252
017240  106760  177377                  MFPS    -401(R0)        ;TRY MFPS W/MODE 6
017244  102403                          BVS     MFPS6A          ;BR TO ERROR IF V-BIT SET
017246  103402                          BCS     MFPS6A          ;BR TO ERROR IF C-BIT SET
017250  001401                          BEQ     MFPS6A          ;BR TO ERROR IF Z-BIT SET
017252  100401                          BMI     MFPS6B
017254                          MFPS6A:
017254  104000                          EMT                     ;COND. CODES INCORRECT
017256  022737  000252  000000  MFPS6B: CMP     #252,@#0        ;CHECK DEST. DATA
017264  001401                          BEQ     MFPS6C
017266  104000                          EMT                     ;DEST. DATA INCORRECT
017270  022700  000401          MFPS6C: CMP     #401,R0         ;CHECK DEST. REGISTER
017274  001401                          BEQ     TS257
017276  104000                          EMT                     ;DEST. DATA INCORRECT

                                ;*****************************************************************************
                                ;TEST 257       TEST MFPS MODE 7
                                ;*****************************************************************************
017300                          TS257:
017300  012700  000777                  MOV     #777,R0         ;R0=777
017304  005037  000000                  CLR     @#0             ;LOC. 0=0
017310  012737  000125  177776          MOV     #125,@#PS       ;PS=125
017316  106770  177407                  MFPS    @-371(R0)       ;TRY MFPS W/MODE 7
017322  102403                          BVS     MFPS7A          ;BR TO ERROR IF V-BIT SET
017324  103002                          BCC     MFPS7A          ;BR TO ERROR IF C-BIT SET
017326  001401                          BEQ     MFPS7A          ;BR TO ERROR IF Z-BIT SET
017330  100001                          BPL     MFPS7B
017332                          MFPS7A:
017332  104000                          EMT                     ;CONDITION CODE INCORRECT
017334  022737  042400  000000  MFPS7B: CMP     #42400,@#0      ;CHECK DESTINATION DATA
017342  001401                          BEQ     MFPS7C
017344  104000                          EMT                     ;DEST. DATA INCORRECT
017346  022700  000777          MFPS7C: CMP     #777,R0         ;CHECK MODE 7 REGISTER
017352  001401                          BEQ     TS260
017354  104000                          EMT                     ;MODE 7 REGISTER MODIFIED

                                ;*****************************************************************************
                                ;
                                ;       THIS TEST VERIFIES THAT RESET DOES NOT CLEAR THE PSW.
                                ;THE PSW IS LOADED WITH ONES, A RESET IS ISSUED, AND THE
                                ;CONTENTS OF THE PSW ARE CHECKED TO VERIFY THAT THEY HAVE NOT
                                ;CHANGED. THIS TEST IS EXECUTED ONLY ONCE EVERY 240 (DECIMAL)
                                ;ITERATIONS OF PROGRAM.
                                ;
                                ;*****************************************************************************
                                ;TEST 260       TEST THAT RESET DOES NOT CLEAR PSW
                                ;*****************************************************************************
017356                          TS260:
017356  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
```

```
                        BEQ     70$             ;IF NO THEN DO TEST
                        TST     @#$PASS         ;IS THIS FIRST PASS
                        BNE     TS261           ;IF NO THEN SHIP TO NEXT TEST
                70$:
017364  001403          MOV     #357,@#PS               ;MOV ONES TO PSW
017366  005737  C01006  RESET                   ;
017372  001011          CMP     #357,@#PS               ;PSW CORRECT?
017374                  BEQ     TS261
017374  012737  000357  EMT                     ;RESET ALTERED PSW
017402  000005
017404  022737  000357
017412  001401
017414  104000
017416                  REST:

                        ;*****************************************************************************************
                        ;
                        ;       THE FOLLOWING TEST CHECKS THE INDEPENDENT FUNCTIONING OF BASIC
                        ;DATA PATH COMPONENTS WITH USER MODE SET.
                        ;
                        ;*****************************************************************************************
                        ;TEST 261       TEST USER MODE R6 CAN HOLD A ONE IN EVERY POSITION
                        ;*****************************************************************************************
017416                  TS261:
017416  052767  140000  BIS     #USRM,PS                ;SET USER MODE
017424  012706  000001  MOV     #1,R6                   ;SET BIT0
017430  000241          CLC                             ;CLEAR C-BIT
017432  006106          USP1:   ROL     R6              ;ROTATE 1 POSITION
017434  103376          BCC     USP1                    ;BR IF NOT ALL DONE
017436  001404          BEQ     USP1A                   ;BR IF NO BITS PICKED
017440  042767  140000  BIC     #USRM,PS                ;CLEAR USER MODE
017446  104000          EMT                     ;USER MODE R6 PICKED A BIT
017450  042767  140000  USP1A:  BIC     #USRM,PS                ;CLEAR USER MODE


                        ;*****************************************************************************************
                        ;
                        ;       THIS TEST CHECKS THE INDEPENDENT FUNCTIONING OF THE USER
                        ;AND KERNEL MODE R6'S. R6 IS SETUP AND ADDRESSED IN EACH
                        ;OF THE TWO MODES TO VERIFY THAT THE TWO R6'S ARE INDEPENDENT
                        ;OF EACH OTHER.
                        ;
                        ;*****************************************************************************************
                        ;TEST 262       TEST INDEPENDENCE OF USER AND KERNEL MODE R6'S
                        ;*****************************************************************************************
017456                  TS262:
017456  052767  140000  BIS     #USRM,PS                ;SET USER MODE
017464  012706  177777  MOV     #-1,R6                  ;SET USER R6 TO ALL ONES
017470  022706  177777  CMP     #-1,R6                  ;READ AND CHECK USER R6
017474  001404          BEQ     USP2                    ;BR IF NO ERROR
017476  042767  140000  BIC     #USRM,PS                ;CLEAR USER MODE
017504  104000          EMT                     ;USER R6 WILL NOT HOLD ALL ONES
017506  042767  140000  USP2:   BIC     #USRM,PS                ;SET KERNEL MODE
017514  022706  177777  CMP     #-1,R6                  ;KERNEL MODE R6 ADDR. FROM USER MODE?>>
017520  001001          BNE     USP3
017522  104000          EMT                     ;DUAL ADDRESSING ERROR USER/KERNEL R6
017524  005006          USP3:   CLR     R6              ;CLEAR KERNEL MODE SP
017526  052767  140000  BIS     #USRM,PS                ;SET USER MODE
017534  022706  177777  CMP     #-1,R6                  ;CHECK USER R6 NOT ADDR. FROM KERNEL MODE
017540  042767  140000  BIC     #USRM,PS                ;CLEAR USER MODE
```

```
5534  017546  001401                          BEQ     USP4                    ;BR IF NO ERROR
5535  017550  104000                          EMT                     ;DUAL ADDRESSING ERROR OR SEQUENCE ERROR
5536  017552  012706  001000          USP4:   MOV     #STBOT,R6               ;RESTORE SP USER
5537  017556  042767  140000  160212          BIC     #USRM,PS                ;SET KERNEL MODE
5538  017564  012706  001000                  MOV     #STBOT,R6               ;RESTORE SP KERNEL
5539
5540                                  ;*****************************************************************************
5541                                  ;
5542                                  ;       THESE NEXT TWO TESTS VERIFY MFPI AND MTPI INSTRUCTIONS
5543                                  ;WITH R6 IN MODE 0.
5544                                  ;
5545                                  ;*****************************************************************************
5546                                  ;TEST 263       TEST MFPI WITH R6 IN MODE 0
5547                                  ;*****************************************************************************
5548  017570                          TS263:
5549  017570  012706  001000                  MOV     #STBOT,R6               ;INITIALIZE KERNEL STACK POINTER
5550  017574  012767  140000  160174          MOV     #USRM,PS                ;SET USER MODE.PREVIOUS KERNEL
5551  017602  012706  000600                  MOV     #USESTK,R6              ;INITIALIZE USER STACK POINTER
5552  017606  006506                          MFPI    R6                      ;TRY MFPI WITH MODE 0
5553  017610  022767  140000  160160          CMP     #140000,PS              ;CHECK PSW
5554  017616  001404                          BEQ     MFPI0                   ;BR IF NO ERROR
5555  017620  042767  140000  160150          BIC     #USRM,PS                ;CLEAR USER MODE
5556  017626  104000                          EMT                     ;INCORRECT PSW FROM MFPI
5557  017630  042767  140000  160140  MFPI0:  BIC     #USRM,PS                ;CLEAR USER MODE
5558  017636  022767  001000  160732          CMP     #STBOT,USESTK-2         ;CHECK DATA ON STACK
5559  017644  001401                          BEQ     MFPI0A                  ;BR IF NO ERROR
5560  017646  104000                          EMT                     ;INCORRECT DATA FROM MFPI
5561  017650                          MFPI0A:
5562
5563                                  ;*****************************************************************************
5564                                  ;TEST 264       TEST MTPI WITH R6 IN MODE 0
5565                                  ;*****************************************************************************
5566  017650                          TS264:
5567  017650  005067  160122                  CLR     PS                      ;SET KERNEL MODE
5568  017654  005006                          CLR     R6                      ;INITIALIZE KERNEL R6
5569  017656  012767  140000  160112          MOV     #USRM,PS                ;SET USER MODE/PREVIOUS KERNEL
5570  017664  012706  000600                  MOV     #USESTK,R6              ;INITIALIZE USER STACK POINTER
5571  017670  012746  001000                  MOV     #STBOT,-(R6)            ;SET UP TARGET DATA
5572  017674  006606                          MTPI    R6                      ;TRY MODE 0 MTPI
5573  017676  022767  140000  160072          CMP     #USRM,PS                ;CHECK PSW
5574  017704  001404                          BEQ     MTPI0                   ;BR IF NO ERROR
5575  017706  042767  140000  160062          BIC     #USRM,PS                ;CLEAR USER MODE
5576  017714  104000                          EMT                     ;PS INCORRECT FOLLOWING MTPI
5577  017716  005067  160054          MTPI0:  CLR     PS                      ;SET KERNEL MODE
5578  017722  020627  001000                  CMP     R6,#STBOT               ;CHECK TARGET DATA
5579  017726  001401                          BEQ     TS265
5580  017730  104000                          EMT                     ;DATA INCORRECT FOLLOWING MTPI
5581
5582
5583
5584                                  ;*****************************************************************************
5585                                  ;
5586                                  ;THE FOLLOWING TEST VERIFIES THAT NO DUAL ADDRESSING OF THE GENERAL
5587                                  ;REGISTERS OCCURS. ALL REGISTERS ARE CLEARED, AND A UNIQUE BIT IS SET
5588                                  ;IN EACH. CMP INSTRUCTIONS CHECK THAT ONLY ONE BIT IS SET IN EACH
5589                                  ;REGISTER.
```

```
                        ;
                        ;*****************************************************************************
                        ;TEST 265       DUAL REGISTER ADDRESSING TEST
                        ;*****************************************************************************
017732                  TS265:
017732  005000          BITCLR: CLR     R0              ;INITIALIZE ALL REGISTERS
017734  005001                  CLR     R1
017736  005002                  CLR     R2
017740  005003                  CLR     R3
017742  005004                  CLR     R4
017744  005005                  CLR     R5
017746  005006                  CLR     R6
017750  052700  000001  BITSET: BIS     #1,R0           ;SET R0=1
017754  052701  000002          BIS     #2,R1           ;R1=2
017760  052702  000004          BIS     #4,R2           ;R2=4
017764  052703  000010          BIS     #10,R3          ;R3=10
017770  052704  000020          BIS     #20,R4          ;R4=20
017774  052705  000040          BIS     #40,R5          ;R5=40
020000  052706  000100          BIS     #100,R6         ;R6=100
020004  022706  000100  BITCHK: CMP     #100,R6         ;TEST THAT NO DUAL ADDRESSING OCCURRED
020010  001022                  BNE     DAERR           ;BR TO ERROR HALT IF ANY OTHER BITS ARE SET
020012  022705  000040          CMP     #40,R5
020016  001017                  BNE     DAERR
020020  022704  000020          CMP     #20,R4
020024  001014                  BNE     DAERR
020026  022703  000010          CMP     #10,R3
020032  001011                  BNE     DAERR
020034  022702  000004          CMP     #4,R2
020040  001006                  BNE     DAERR
020042  022701  000002          CMP     #2,R1
020046  001003                  BNE     DAERR
020050  022700  000001          CMP     #1,R0
020054  001401                  BEQ     BITCON
020056                  DAERR:
020056  104000                  EMT                     ;DUAL ADDRESSING ERROR
020060  012702  001004  BITCON: MOV     #$TESTN,R2      ;RESTORE POINTER

                        ;*****************************************************************************
                        ;       THIS TEST VERIFIES THAT THE UPPER BYTE OF THE PSW IS NOT AFFECTED
                        ;WHEN THE PRIORITY LEVEL OR CC'S ARE CHANGED. ALL BITS ARE
                        ;INITIALLY SET IN THE PSW, AND THE LOW BYTE IS CLEARED. A BIT
                        ;INSTRUCTION VERIFIES THE DATA.
                        ;
                        ;*****************************************************************************
                        ;TEST 266       TEST BYTE INSTRUCTION ON PSW
                        ;*****************************************************************************
020064                  TS266:
020064  052737  170357  177776          BIS     #170357,@#PS    ;SET ALL POSSIBLE BITS IN PSW
020072  105037  177776                  CLRB    @#PS            ;CLR PR LEVEL AND CC'S
020076  013700  177776                  MOV     @#PS,R0         ;COPY CONTENTS OF PSW
020102  032700  170000                  BIT     #170000,R0      ;TEST THAT UPPER BYTE IS UNAFFECTED
020106  001003                          BNE     BTCON           ;CONTINUE IF OK
020110  005037  177776  BTERR:  CLR     @#PS            ;RETURN TO KERNEL MODE
020114  104000                  EMT                     ;BYTE INSTRUCTION ALTERED PSW
020116  005037  177776  BTCON:  CLR     @#PS            ;RETURN TO KERNEL MODE
```

```
5646
5647                    ;*****************************************************************************
5648                    ;
5649                    ;       THIS TEST VERIFIES THAT A JMP INSTRUCTION DOES NOT ALTER THE
5650                    ;CONDITION CODES IN THE PSW. THE CC'S ARE PRESET,THE JMP IS
5651                    ;EXECUTED, AND CONDITIONAL BRANCHES VERIFY THE STATE OF THE CC'S.
5652                    ;
5653                    ;*****************************************************************************
5654                    ;TEST 267       TEST THAT JMP INSTRUCTION DOES NOT AFFECT CONDITION CODES
5655 020122            ;*****************************************************************************
5656 020122 000277  TS267:  SCC
5657 020124 000252          +CLN!CLV                ;CC=0101
5658 020126 000167 000000   JMP     JMPT            ;JUMP TO TEST PSW
5659 020132 100403  JMPT:   BMI     JMPERR          ;BR TO ERROR HALT IF N-BIT IS SET
5660 020134 001002          BNE     JMPERR          ;BR TO ERROR HALT IF Z-BIT IS CLEAR
5661 020136 102401          BVS     JMPERR          ;BR TO ERROR HALT IF V-BIT IF SET
5662 020140 103401          BCS     TS270
5663 020142         JMPERR:
5664 020142 104000          EMT                     ;JMP INSTRUCTION AFFECTED CC'S
5665                    ;*****************************************************************************
5666                    ;
5667                    ;       THIS TEST VERIFIES THE SET AND CLEAR CONDITION CODE INSTRUCTIONS.
5668                    ;THE TEST CONSISTS OF TWO ROUTINES, ONE TO TEST ALL CLEAR CC
5669                    ;INSTRUCTIONS, AND THE SECOND TO TEST ALL SET CC INSTRUCTIONS. ALL
5670                    ;POSSIBLE COMBINATIONS OF CONDITION CODES ARE TESTED, INCLUDING NOP'S.
5671                    ;       TO TEST THE CLEAR CC INSTRUCTIONS, ALL CONDITION CODES ARE
5672                    ;INITIALLY SET. THE INSTRUCTION IS EXECUTED, AND THE PSW IS CHECKED
5673                    ;TO VERIFY THE PROPER COMBINATION OF CONDITION CODES.
5674                    ;       TO TEST THE SET CC INSTRUCTIONS, THE CONDITION CODES ARE
5675                    ;INITIALLY CLEARED, AND ONLY THE REQUIRED BITS ARE SET BY THE SET CC
5676                    ;INSTRUCTION. THE CONTENTS OF THE PSW ARE CHECKED TO VERIFY THAT
5677                    ;ONLY THE REQUIRED BITS WERE SET.
5678                    ;
5679                    ;*****************************************************************************
5680                    ;TEST 270       TEST SET CC AND CLEAR CC INSTRUCTIONS
5681                    ;*****************************************************************************
5682 020144         TS270:
5683 020144 012767 000240 000024        MOV     #240,CC3        ;INITIALIZE CLR CC INSTRUCTION CODES
5684 020152 012767 000017 000032        MOV     #17,CC2         ;INITIALIZE OCTAL MAP
5685 020160 012767 000261 000074        MOV     #261,SC3        ;INITIALIZE SET CC INSTRUCTION CODES
5686 020166 012767 000001 000102        MOV     #1,SC4          ;INITIALIZE OCTAL MAP
5687 020174 000277  CLRCD:  SCC                         ;SET ALL CONDITION CODES
5688 020176 000000  CC3:    0                           ;CONDITION CODE INSTRUCTION
5689 020200 013704 177776               MOV     @#PS,R4         ;COPY THE PSW
5690 020204 042704 177760               BIC     #177760,R4      ;ISOLATE CONDITION CODES
5691 020210 022704                      CMP     (PC)+,R4        ;CHECK THAT PROPER CC'S WERE CLEARED
5692 020212 000000  CC2:    0                           ;OCTAL REPRESENTATION OF CC'S
5693 020214 001401                      BEQ     CON1
5694 020216 104000                      EMT                     ;CLEAR CC INSTRUCTION FAILED
5695 020220 005367 177766       CON1:   DEC     CC2             ;SET NEXT OCTAL MAP OF CC'S
5696 020224 005267 177746               INC     CC3             ;GET NEXT CLEAR CC INSTRUCTION
5697 020230 026727 177742 000257        CMP     CC3,#257        ;TEST FOR CCC INSTRUCTION
5698 020236 003756                      BLE     CLRCD           ;GO TEST NEXT INSTRUCTION IF NOT FOUND
5699 020240 026727 177732 000260        CMP     CC3,#260        ;CHECK FOR NOP=260
5700 020246 001004                      BNE     SETCD           ;GO TEST SET CC INSTRUCTIONS
5701 020250 012767 000017 177734        MOV     #17,CC2         ;SET OCTAL MAP TO TEST NOP
```

```
5702  020256  000746                        BR      CLRCD           ;GO TEST NOP
5703  020260  000257                SETCD:  CCC                     ;CLEAR ALL CONDITION CODES
5704  020262  000000                SC3:    0                       ;CONDITION CODE INSTRUCTION
5705  020264  013704  177776                MOV     @#PS,R4         ;COY PSW
5706  020270  042704  177760                BIC     #177760,R4      ;CLEAR AWAY UNWANTED BITS
5707  020274  022704                        CMP     (PC)+,R4        ;CHECK THAT PROPER CC'S WERE SET
5708  020276  000000                SC4:    0                       ;OCTAL REPRESENTATION OF CC'S
5709  020300  001401                        BEQ     CON2
5710  020302                        CCERR:
5711  020302  104000                        EMT                     ;SET CC FAILED OR SEQUENCE ERROR
5712  020304  005267  177766        CON2:   INC     SC4             ;SET NEXT OCTAL MAP
5713  020310  005267  177746                INC     SC3             ;PREPARE NEXT SET CC INSTRUCTION
5714  020314  026727  177742  000277        CMP     SC3,#277        ;FINISHED?
5715  020322  003756                        BLE     SETCD           ;BR IF NO
5716  020324  000167  000006                JMP     MORO            ;JUMP TO NEXT TESTS
```

```
                        ;*****************************************************************************
                        .SBTTL  TEST INSTRUCTIONS USING SAME REGISTER FOR SOURCE & DESTINATION
                        ;
                        ;IN AUTO INCREMENT (DECREMENT) MODES AND
                        ;AUTO INCREMENT (DECREMENT) DEFERRED MODES,
                        ;CONTENTS OF THE REGISTER IN USED ARE
                        ;INCREMENTED (DECREMENTED) BY 2
                        ;BEFORE USED AS THE SOURCE OPERAND.
                        ;
020330  000000  000000  000000  A:      .WORD 0,0,0
020336                  MOR0:
                        ;*****************************************************************************
                        ;TEST 271       TEST AUTO-INCREMENT MODE, USING R0
                        ;*****************************************************************************
020336                  TS271:
020336  005037  020330          CLR     @#A             ;CLEAR LOC A
020342  012700  020330          MOV     #A,R0           ;R0 STORES ADDR OF A
020346  060020                  ADD     R0,(R0)+        ;CHECK THAT R0 IS INCR BY 2 BEFORE
                                                        ;BEING USED AS THE SOURCE OPERAND
020350  022700  020332          CMP     #A+2,R0         ;R0 INCR BY 2?
020354  001401                  BEQ     MOR1
020356  104000                  EMT                     ;R0 WAS NOT INCREMENTED BY 2
                                                        ;
020360  022737  020332  020330  MOR1:   CMP     #A+2,@#A        ;CHECK CONTENT OF R0 WAS INCR BY 2 BEFORE
                                                        ;BEING USED IN THE "ADD" INSTR
                                                        ;LOC A CONTAINS (A+2)?
020366  001401                  BEQ     TS272
020370  104000                  EMT                     ;WRONG SUM IN LOC A
                        ;*****************************************************************************
                        ;TEST 272       AUTO-DECREMENT MODE, USING R0
                        ;*****************************************************************************
020372                  TS272:
020372  005037  020330          CLR     @#A             ;CLEAR LOC A
020376  012700  020332          MOV     #A+2,R0         ;R0 STORES ADDR OF A+2
020402  060040                  ADD     R0,-(R0)        ;CHECK THAT R0 IS DECR BY 2 BEFORE
                                                        ;BEING USED AS THE SOURCE OPERAND
020404  022700  020330          CMP     #A,R0           ;R0 DECR BY 2?
020410  001401                  BEQ     MOR2
020412  104000                  EMT                     ;R0 WAS NOT DECREMENTED BY 2
                                                        ;
020414  022737  020330  020330  MOR2:   CMP     #A,@#A          ;CONTENT OF R0 WAS DECR BY 2 BEFORE
                                                        ;BEING USED IN THE "ADD" INSTR
                                                        ;LOC A CONTAINS (R0)
020422  001401                  BEQ     TS273
020424  104000                  EMT                     ;WRONG SUM IN LOC A
                                                        ;
                        ;*****************************************************************************
                        ;TEST 273       TEST AUTO-INCREMENT DEFERRED MODE, USING R0
                        ;*****************************************************************************
020426                  TS273:
020426  005037  020330          CLR     @#A             ;CLEAR LOC A
020432  005037  020334          CLR     @#A+4           ;CLEAR LOC A+4
020436  012737  020330  020332  MOV     #A,@#A+2        ;STORE ADDR A IN LOC A+2
020444  012700  020332          MOV     #A+2,R0         ;R0 STORES ADDR A+2
020450  060030                  ADD     R0,@(R0)+       ;CHECK THAT R0 IS INCR BY 2 BEFORE
```

```
5773
5774  020452  022700  020334              CMP     #A+4,R0         ;BEING USED AS THE SOURCE OPERAND
5775  020456  001401                      BEQ     MOR3            ;R0 INCR BY 2?
5776  020460  104000                      EMT                     ;R0 WAS NOT INCREMENTED BY 2
5777                                                              ;
5778  020462  022737  020330  020332 MOR3: CMP     #A,@#A+2        ;LOC A+2 STILL STORES ADDR A?
5779  020470  001401                      BEQ     MOR4
5780  020472  104000                      EMT                     ;LOC A+2 STORES WRONG DATA
5781                                                              ;
5782  020474  022737  020334  020330 MOR4: CMP     #A+4,@#A        ;CHECK CONTENT OF R0 WAS INCR BY 2 BEFORE
5783                                                              ;BEING USED IN THE ''ADD'' INSTR
5784  020502  001401                      BEQ     MOR5
5785  020504  104000                      EMT                     ;LOC A STORES WRONG DATA
5786                                                              ;
5787  020506  005737  020334         MOR5: TST     @#A+4           ;LOC A+4 STILL STORES 0?
5788  020512  001401                      BEQ     TS274
5789  020514  104000                      EMT                     ;LOC A+4 DID NOT STAY CLEAR
5790
5791                                  ;*********************************************************************************
5792                                  ;TEST 274       TEST AUTO-DECREMENT DEFERRED, USING R0
5793                                  ;*********************************************************************************
5794  020516                          TS274:
5795  020516  005037  020330              CLR     @#A             ;CLEAR LOC A
5796  020522  005037  020334              CLR     @#A+4           ;CLEAR LOC A+4
5797  020526  012700  020334              MOV     #A+4,R0         ;R0 STORES ADDR A+4
5798  020532  012737  020330  020332      MOV     #A,@#A+2        ;STORE ADDR A IN LOC A+2
5799  020540  060050                      ADD     R0,@-(R0)       ;CHECK THAT R0 IS DECR BY 2 BEFORE
5800                                                              ;BEING USED AS THE SOURCE OPERAND
5801  020542  022700  020332              CMP     #A+2,R0         ;R0 DECREMENTED BY 2?
5802  020546  001401                      BEQ     MOR6
5803  020550  104000                      EMT                     ;R0 WAS NOT DECREMENTED BY 2
5804                                                              ;
5805  020552  022737  020332  020330 MOR6: CMP     #A+2,@#A        ;CHECK CONTENT OF R0 WAS DECR BY 2 BEFORE
5806                                                              ;BEING USED IN THE ''ADD'' INSTR
5807  020560  001401                      BEQ     MOR7
5808  020562  104000                      EMT                     ;LOC A STORES WRONG DATA
5809
5810                                                              ;
5811  020564  022737  020330  020332 MOR7: CMP     #A,@#A+2        ;LOC A+2 STILL STORES A?
5812  020572  001401                      BEQ     MOR8
5813  020574  104000                      EMT                     ;LOC A+2 STORES WRONG DATA
5814                                                              ;
5815  020576  005737  020334         MOR8: TST     @#A+4           ;LOC A+4 STILL STORES 0?
5816  020602  001401                      BEQ     TS275
5817  020604  104000                      EMT                     ;LOC A+4 DID NOT STAY CLEAR
5818                                                              ;
```

```
5819                                    ;*****************************************************************************
5820                                    .SBTTL  INSTRUCTION USING PC AS SOURCE REGISTER
5821                                    ;
5822                                    ;IN INDEX, INDEX DEFERRED, RELATIVE, AND
5823                                    ;RELATIVE DEFERRED MODES, DESTINATION WILL CONTAIN
5824                                    ;THE PC COUNT OF THE CURRENT INSTRUCTION +4.
5825                                    ;
5826                                    ;*****************************************************************************
5827                                    ;TEST 275       TEST PC AS SOURCE IN MODE 0, USING R0
5828                                    ;*****************************************************************************
5829  020606                            TS275:
5830  020606  012700  177777                   MOV     #-1,R0          ;SET ALL 1 IN R0
5831  020612  010700                    PCN01:  MOV     PC,R0           ;STORES PC IN R0
5832  020614  022700  020614                   CMP     #PCN01+2,R0     ;R0 STORES PC+2?
5833  020620  001401                           BEQ     TS276
5834  020622  104000                           EMT                     ;R0 STORED WRONG VALUE
5835
5836                                    ;*****************************************************************************
5837                                    ;TEST 276       TEST PC AS SOURCE IN MODE 6, USING R0
5838                                    ;*****************************************************************************
5839  020624                            TS276:
5840  020624  012700  020330                   MOV #A,R0               ;R0 STORES ADDR A
5841  020630  010760  000004            PCN2:  MOV PC,4(R0)            ;EFFECTIVE ADDR IS A+4
5842  020634  022737  020634  020334            CMP     #PCN2+4,@#A+4   ;LOC A+4 STORES PC+4?
5843  020642  001401                           BEQ     TS277
5844  020644  104000                           EMT                     ;LOC A+4 STORED WRONG VALUE
5845
5846                                    ;*****************************************************************************
5847                                    ;TEST 277       TEST PC AS SOURCE IN MODE 7, USING R0
5848                                    ;*****************************************************************************
5849  020646                            TS277:
5850  020646  012737  020330  020334            MOV     #A,@#A+4        ;LOC A+4 STORES ADDR A
5851  020654  012700  020330                    MOV     #A,R0           ;R0 STORES ADDR A
5852  020660  010770  000004            PCN3:   MOV     PC,@4(R0)       ;EFFECTIVE ADDR IS A
5853  020664  022737  020664  020330            CMP     #PCN3+4,@#A     ;LOC A STORES PC+4?
5854  020672  001401                           BEQ     TS300
5855  020674  104000                           EMT                     ;LOC A STORED WRONG VALUE
5856
5857                                    ;*****************************************************************************
5858                                    ;TEST 300       TEST PC AS SOURCE IN RELATIVE DEFERRED MODE ,USING R0
5859                                    ;*****************************************************************************
5860  020676                            TS300:
5861  020676  012737  020332  020330            MOV     #A+2,@#A        ;LOC A STORES ADDR A+2
5862  020704  010777  177420            PCN4:   MOV     PC,@A           ;EFFECTIVE ADDR IS A+2
5863  020710  022737  020710  020332            CMP     #PCN4+4,@#A+2   ;LOC A+2 STORES PC+4?
5864  020716  001401                           BEQ     TS301
5865  020720  104000                           EMT                     ;LOC A+2 STORED WRONG VALUE
5866
5867                                    ;*****************************************************************************
5868                                    ;TEST 301       TEST PC AS SOURCE IN RELATIVE MODE ,USING R0
5869                                    ;*****************************************************************************
5870  020722                            TS301:
5871  020722  005037  020330                    CLR     @#A             ;CLEAR A
5872  020726  010767  177376            PCN5:   MOV     PC,A            ;EFFECTIVE ADDR IS A
5873  020732  022737  020732  020330            CMP     #PCN5+4,@#A     ;LOC A STORES PC+4?
5874  020740  001401                           BEQ     TS302
```

```
5875  020742  104000                          EMT                             ;LOCATION A STORED WRONG VALUE
5876                                  ;*************************************************************************
5877                                  ;THIS TESTS THE MOVE FROM PROCESSOR TYPE INSTRUCTION(MFPT)
5878                                  ;UPON EXECUTION R0 WILL RECIEVE THE PROCESSOR MODEL CODE
5879                                  ;WHICH IS ''000003'' FOR THE DCF11-AA
5980                                  ;*************************************************************************
5881                                  ;*************************************************************************
5882                                  ;TEST 302       TEST MFPT
5883                                  ;*************************************************************************
5884  020744                          TS302:
5885          000007                  MFPT=000007
5886  020744  012706  001000                 MOV     #STBOT,SP       ;INITIALIZE STACK POINT IN CASE OF TRAP
5887  020750  000007                         MFPT                    ;GET MODEL CODE.IF THIS TRAPS AN ERROR WILL BE REPORTED
5888  020752  022700  000003                 CMP     #3,R0           ;CHECK IF CORRECT CODE RETURNED
5889  020756  001401                         BEQ     TS303
5890  020760  104000                         EMT                     ;WRONG CODE RETURNED
5891
5892
5893                                  ;*************************************************************************
5894                                  .SBTTL  THE NEXT THREE TESTS EXERCISE MASKING ACTION OF MICROCODES.
5895                                  ;*************************************************************************
5896                                  ;TEST 303       TEST SUB INSTRUCTION, SM=0, DM=2
5897                                  ;*************************************************************************
5898                                  TS303:
5899  020762
5900  020762  012737  052525  000000          MOV     #052525,@#0     ;SET UP LOC 0
5901  020770  012701  050505                 MOV     #050505,R1      ;SET UP R1
5902  020774  005000                         CLR     R0              ;CLEAR R0
5903  020776  160120                         SUB     R1,(R0)+        ;SUBTRACTION, SM=0,DM=2
5904  021000  022737  002020  000000         CMP     #2020,@#0       ;CHECK DIFFERENCE AT LOC 0
5905  021006  001401                         BEQ     TS304
5906  021010  104000                         EMT                     ;WRONG RESULT FROM SUBTRACTION
5907
5908                                  ;*************************************************************************
5909                                  ;TEST 304       TEST MFPD WITH R0, IN MODE 2
5910                                  ;*************************************************************************
5911  021012                          TS304:
5912  021012  012737  052525  000000          MOV     #052525,@#0     ;SET UP LOC 0
5913  021020  005000                         CLR     R0              ;CLEAR R0
5914  021022  012767  170000  156746         MOV     #170000,PS      ;SET USER MODE ON, CURRENT & PREVIOUS
5915  021030  012706  000600                 MOV     #USESTK,R6      ;SET USER STACK POINTER
5916  021034  106520                         MFPD    (R0)+           ;MODE 2, MFPD
5917  021036  005067  156734                 CLR     PS              ;SET KERNEL MODE
5918  021042  022767  052525  157526         CMP     #052525,USESTK-2        ;CHECK DATA ON STACK
5919  021050  001401                         BEQ     TS305
5920  021052  104000                         EMT                     ;INCORRECT DATA FROM MFPD
5921
5922                                  ;*************************************************************************
5923                                  ;TEST 305       TEST MTPD WITH R0, IN MODE 2
5924                                  ;*************************************************************************
5925  021054                          TS305:
5926  021054  012767  170000  156714         MOV     #170000,PS      ;SET USER MODE ON, CURRENT & PREVIOUS
5927  021062  012706  000600                 MOV     #USESTK,R6      ;SET USER STACK POINTER
5928  021066  012746  125252                 MOV     #125252,-(R6)   ;PUSH DATA IN USER STACK
5929  021072  012737  000000  000000         MOV     #0,@#0          ;CLEAR LOC 0
5930  021100  005000                         CLR     R0              ;CLEAR R0
```

```
5931  021102  106620                  MTPD    (R0)+           ;MODE 2, MTPD
5932  021104  005067  156666          CLR     PS              ;SET KERNEL MODE
5933  021110  022737  125252  000000  CMP     #125252,@#0     ;CHECK DATA ON LOC 0
5934  021116  001463                  BEQ     TESTN1
5935  021120  104000                  EMT                     ;INCORRECT DATA FROM MTPD
5936
5937  021122  000402          BRTAB:  BR      .+6
5938  021124  001002                  BNE     .+6
5939  021126  001402                  BEQ     .+6
5940  021130  002002                  BGE     .+6
5941  021132  002402                  BLT     .+6
5942  021134  003002                  BGT     .+6
5943  021136  003402                  BLE     .+6
5944  021140  100002                  BPL     .+6
5945  021142  100402                  BMI     .+6
5946  021144  101002                  BHI     .+6
5947  021146  101402                  BLOS    .+6
5948  021150  102002                  BVC     .+6
5949  021152  102402                  BVS     .+6
5950  021154  103002                  BCC     .+6             ;SAME AS BHIS
5951  021156  103402                  BCS     .+6             ;SAME AS BLO
5952
5953          000002          .RADIX  2
5954  021160  177777          YNTAB:  1111111111111111        ;BR
5955  021162  170360                  1111000011110000        ;BNE:  Z=0
5956  021164  007417                  0000111100001111        ;BEQ:  Z=1
5957  021166  146063                  1100110000110011        ;BGE:  N XOR V =0
5958  021170  031714                  0011001111001100        ;BLT:  N XOR V =1
5959  021172  140060                  1100000000110000        ;BGT:  Z+(N XOR V) =0
5960  021174  037717                  0011111111001111        ;BLE:  Z+(N XOR V) =1
5961
5962  021176  177400                  1111111100000000        ;BPL:  N=0
5963  021200  000377                  0000000011111111        ;BMI:  N=1
5964  021202  120240                  1010000010100000        ;BHI:  C+Z=0
5965  021204  057537                  0101111101011111        ;BLOS: C+Z=1
5966  021206  146314                  1100110011001100        ;BVC:  V=0
5967  021210  031463                  0011001100110011        ;BVS:  V=1
5968  021212  125252                  1010101010101010        ;BCC:  C=0
5969  021214  052525                  0101010101010101        ;BCS:  C=1
5970          000010          .RADIX  8
5971
5972                          ;*****************************************************************************
5973                          ; THE FOLLOWING ARE SPECIAL CPU TRAP
5974                          ;HANDLERS TO TRAP AND REPORT SPECIAL TRAPS.
5975                          ;
5976                          ;*****************************************************************************
5977
5978  021216                  T04:
5979  021216  104000                  EMT                     ;TRAPPED THRU LOC. 4
5980  021220                  T010:
5981  021220  104000                  EMT                     ;TRAPPED THRU LOC. 10
5982  021222                  T014:
5983  021222  104000                  EMT                     ;TRAPPED THRU LOC. 14
5984  021224                  T020:
5985  021224  104000                  EMT                     ;TRAPPED THRU LOC. 20
5986  021226                  T030:
```

```
                        EMT                     ;TRAPPED THRU LOC. 30
TO34:
                        EMT                     ;TRAPPED THRU LOC. 34
TO114:
                        EMT                     ;TRAPPED THRU LOC. 114
TO244:
                        EMT                     ;TRAPPED THRU LOC. 244
TO250:
                        EMT                     ;TRAPPED THRU LOC. 250
.SBTTL ** STARTING OF TRAP TEST **

;SPECIAL CASE OF ODD;.EVEN .BYTE AND REGISTER 6
HERE=0

K1:     0
K2:     0
K3:     0
K4:     0
K5:     0
K6:     0
K7:     052525
K10:    052400
K11:    0
K12:    0
SWR:    176

TESTN1: BIT     #1,@#$ENV
        BEQ     1$
        MOV     #$SWREG,SWR
1$:
;*****************************************************************************************
;TEST 306       TEST AUTO INCREMENT AND DECREMENT OF R6 FOR WORD AND BYTES
;*****************************************************************************************
TS306:
        CLR     %6
        MOVB    (6)+,HERE       ;SIX SHOULD INCREMENT BY TWO
        CMP     %6,#2
        BEQ     BR1
        EMT                     ;R6 DID NOT AUTO INCREMENT BY TWO

BR1:    MOV     #1000,%6
        MOVB    -(6),#HERE      ;SHOULD DECREMENT BY TWO
        CMP     %6,#776
        BEQ     BR2
        EMT                     ;R6 DID NOT AUTO DECREMENT BY 2

BR2:    CLR     %6
        MOVB    (6)+,(6)+       ;DOUBLES AUTO INCREMENT OF R6
        CMP     %6,#4
        BEQ     BR3
        EMT                     ;WRONG AUTO INCREMENT OF R6

BR3:    CLR     %6
        CLR     %4
        CMPB    (6)+,(4)+       ;TEST INCREMENT OF R6
        CMP     %6,#2
```

```
6043  021370  001401                        BEQ     BR4
6044  021372  104000                        EMT                     ;WRONG INCREMENT OF R6
6045
6046  021374  005006                BR4:    CLR     %6
6047  021376  005004                        CLR     %4
6048  021400  122426                        CMPB    (4)+,(6)+       ;TEST INCREMENT OF R6
6049  021402  020627  000002                CMP     %6,#2
6050  021406  001401                        BEQ     BR5
6051  021410  104000                        EMT                     ;WRONG INCREMENT OF R6
6052
6053  021412  005006                BR5:    CLR     %6
6054  021414  005004                        CLR     %4
6055  021416  122624                        CMPB    (6)+,(4)+       ;TEST INCREMENT OF R4
6056  021420  020427  000001                CMP     %4,#1
6057  021424  001401                        BEQ     BR6
6058  021426  104000                        EMT                     ;WRONG INCREMENT OF R4
6059  021430  005006                BR6:    CLR     %6
6060  021432  005004                        CLR     %4
6061  021434  122426                        CMPB    (4)+,(6)+       ;TEST INCREMENT OF R6
6062  021436  020627  000002                CMP     %6,#2
6063  021442  001401                        BEQ     BR7
6064  021444  104000                        EMT                     ;WRONG INCREMENT OF R6
6065
6066  021446  005006                BR7:    CLR     %6
6067  021450  005004                        CLR     %4
6068  021452  122426                        CMPB    (4)+,(6)+       ;TEST INCREMENT OF R4
6069  021454  020427  000001                CMP     %4,#1
6070  021460  001401                        BEQ     BR10
6071  021462  104000                        EMT                     ;WRONG INCREMENT OF R4
6072
6073  021464  012706  001000        BR10:   MOV     #1000,%6
6074  021470  124627  000000                CMPB    -(6),#HERE      ;TEST DECREMENT OF R6
6075  021474  022706  000776                CMP     #776,%6
6076  021500  001401                        BEQ     TS307
6077  021502  104000                        EMT                     ;WRONG DECREMENT OF R6,OR WRONG $TSTNM
6078                        ;*****************************************************************************
6079                        ;TEST 307       TEST TRANSFER OF .BYTE USING R6
6080                        ;*****************************************************************************
6081  021504                TS307:
6082  021504  012767  123456  177536        MOV     #123456,K5
6083  021512  012767  050505  177520        MOV     #050505,K1
6084  021520  012705  021240                MOV     #K1,%5          ;%5=(050505)K1
6085  021524  012706  021250                MOV     #K5,%6          ;%6=(123456)K5
6086  021530  112625                        MOVB    (6)+,(5)+       ;LOW .BYTE OF R6 TO R5
6087  021532  022767  050456  177500        CMP     #050456,K1
6088  021540  001401                        BEQ     BR11
6089  021542  104000                        EMT                     ;FALSE TRANSFER OF .BYTE
6090
6091  021544  012767  123456  177476  BR11: MOV     #123456,K5
6092  021552  012767  050505  177460        MOV     #050505,K1
6093  021560  012705  021240                MOV     #K1,%5          ;%5(050505)K1
6094  021564  012706  021252                MOV     #K6,%6          ;%6(123456)K5
6095  021570  114625                        MOVB    -(6),(5)+       ;LOW .BYTE OF R6 TO R5 (DECREMENT)
6096  021572  026727  177442  050456        CMP     K1,#050456
6097  021600  001401                        BEQ     BR12
6098  021602  104000                        EMT                     ;FALSE R6 .BYTE TRANSFER
```

```
6099
6100  021604  012767  123456  177426  BR12:   MOV     #123456,K1
6101  021612  012767  050505  177430          MOV     #050505,K5
6102  021620  012705  021240                  MOV     #K1,%5          ;(123456)
6103  021624  012706  021250                  MOV     #K5,%6          ;(050505)
6104  021630  112526                          MOVB    (5)+,(6)+       ;LOW OF R5 TO LOW OF R6
6105  021632  022767  050456  177410          CMP     #050456,K5
6106  021640  001401                          BEQ     BR13
6107  021642  104000                          EMT                     ;FALSE R6 .BYTE TRANSFER
6108
6109  021644  012767  123456  177366  BR13:   MOV     #123456,K1
6110  021652  012767  050505  177370          MOV     #050505,K5
6111  021660  012705  021241                  MOV     #K1+1,%5        ;123456
6112  021664  012706  021250                  MOV     #K5,%6          ;050505
6113  021670  112526                          MOVB    (5)+,(6)+       ;HIGH OF R5 TO LOW OF R6
6114  021672  026727  177352  050647          CMP     K5,#050647
6115  021700  001401                          BEQ     BR14
6116  021702  104000                          EMT                     ;FALSE R6 .BYTE TRANSFER
6117
6118  021704  012767  123456  177326  BR14:   MOV     #123456,K1
6119  021712  012767  050505  177330          MOV     #050505,K5
6120  021720  012705  021241                  MOV     #K1+1,%5        ;R5-123456-ODD ADDRESS
6121  021724  012706  021250                  MOV     #K5,%6          ;R6-050505--.EVEN ADDRESS
6122  021730  112625                          MOVB    (6)+,(5)+       ;LOW OF R6 TO HIGH OF R5
6123  021732  022767  042456  177300          CMP     #042456,K1
6124  021740  001401                          BEQ     TS310
6125  021742  104000                          EMT                     ;FAILED LOW OF 6 TO HIGH OF 5,OR WRONG $TSTNM
6126                        ;*****************************************************************************
6127                        ;TEST 310       TEST BYTE OPERATION WITH SEQUENTIAL ODD-EVEN ADDRESS
6128                        ;*****************************************************************************
6129  021744                TS310:
6130  021744  126767  177304  177303          CMPB    K7,K7+1         ;SAME .WORD LOW TO HIGH
6131  021752  001401                          BEQ     BR15
6132  021754  104000                          EMT                     ;SHOULD COMPARE LOW TO HIGH
6133
6134  021756  126767  177273  177270  BR15:   CMPB    K7+1,K7         ;COMPARE ODD TO .EVEN SAME .WORD
6135  021764  001401                          BEQ     BR16
6136  021766  104000                          EMT                     ;ODD TO .EVEN .BYTE FAILURE
6137
6138  021770  126767  177263  177256  BR16:   CMPB    K10+1,K7        ;SEQUENTIAL .BYTES
6139  021776  001401                          BEQ     BR17
6140  022000  104000                          EMT                     ;ODD TO .EVEN FAILED
6141
6142  022002  126767  177250  177242  BR17:   CMPB    K10,K6
6143  022010  001401                          BEQ     BR20
6144  022012  104000                          EMT                     ;.EVEN TO EVEN FAILED
6145  022014  126767  177235  177235  BR20:   CMPB    K7+1,K10+1
6146  022022  001401                          BEQ     BR21
6147  022024  104000                          EMT                     ;ODD TO ODD FAILED
6148
6149  022026  126767  177224  177223  BR21:   CMPB    K10,K10+1
6150  022034  001001                          BNE     BR22
6151  022036  104000                          EMT                     ;LOW TO HIGH IN SAME .WORD FAILED
6152
6153  022040  126767  177213  177211  BR22:   CMPB    K10+1,K10+1
6154  022046  001401                          BEQ     BR23
```

```
                        EMT                     ;HIGH TO LOW IN SAME .WORD FAILED

                BR23:   CMPB    K10,K7+1
                        BNE     TS311
                        EMT                     ;.EVEN TO ODD FAILED,OR WRONG $TSTNM

;*****************************************************************************************
;TEST 311       TEST THAT DECREMENT R6 TO A VALUE LESS THAN 400 TRAPS
;*****************************************************************************************
TS311:
                        MOV     #150,%6         ;R6 = 150
                        MOV     #TDEC1,4        ;STACK OVERFLOW TRAP POINTER
                        TST     -(6)            ;WITH R6 = 150 SHOULD TRAP
                        EMT                     ;SHOULD HAVE TRAPPED,OR WRONG $TSTNM
TDEC1:

;*****************************************************************************************
;TEST 312       TEST FOR DECREMENT OF R6 ON OVERFLOW TRAP
;*****************************************************************************************
TS312:
                        MOV     #150,%6         ;R6 = 150
                        MOV     #TDEC2,4        ;TRAP POINTER
                        TST     -(6)            ;WITH R6 = 150 SHOULD TRAP
                TDEC2:  CMP     %6,#142         ;DID R6 DECREMENT
                        BEQ     TS313
                        EMT                     ;R6 NOT = 142,OR WRONG $TSTNM

;*****************************************************************************************
;TEST 313       TEST DIFFERENT TYPES OF OVERFLOW
;*****************************************************************************************
TS313:
                        MOV     #150,%6
                        CLR     146             ;STATUS WORD OF LOC 10
                        MOV     #TDEC3,4        ;RETURN TO LOC 4
                        INC     -(6)
                TDEC3:  TST     146
                        BNE     1$
                        EMT                     ;INCREMENT OPERATION NOT INHIBITED
                1$:     MOV     #1000,%5
                        MOV     #400,%6
                        MOV     #TDEC4,4
                        CMPB    -(6),-(5)
                        EMT                     ;STACK = 400 AND DECREMENTED, SHOULD TRAP
                TDEC4:  MOV     #400,%6
                        MOV     #TDEC7,4
                        BITB    -(5),-(6)
                TDEC6:
                        EMT                     ;NO STACK OVERFLOW,OR WRONG $TSTNM
                TDEC7:

;*****************************************************************************************
;TEST 314       TEST THAT AN 77 CAUSES AN OVERFLOW TRAP
;*****************************************************************************************
TS314:
                        MOV     #400,%6         ;SET UP STACK TO OVERFLOW
```

```
6211  022222  012767  022240  155560          MOV     #VDEC2,10       ;SET UP 77 VECTOR
6212  022230  012767  022244  155546          MOV     #VDEC,4 ;SET UP OVERFLOW VECTOR
6213  022236  000077                          77                      ;THIS TRAP SHOULD CAUSE OVERFLOW
6214  022240  000167  157616          VDEC2:  JMP     ERROR1          ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
6215  022244  012767  021220  155536  VDEC:   MOV     #T010,10        ;RESTORE VECTOR
6216                                  ;*****************************************************************************
6217                                  ;TEST 315       TEST THAT AN IOT CAUSES AN OVERFLOW TRAP
6218                                  ;*****************************************************************************
6219  022252                          TS315:
6220  022252  012706  000400                  MOV     #400,%6         ;SET UP STACK TO OVERFLOW
6221  022256  012767  022274  155534          MOV     #VDEC4,20       ;SET UP IOT VECTOR
6222  022264  012767  022300  155512          MOV     #VDEC3,4        ;SET UP OVERFLOW VECTOR
6223  022272  000004                          IOT                     ;THIS TRAP SHOULD CAUSE OVERFLOW
6224  022274  000167  157562          VDEC4:  JMP     ERROR1          ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
6225  022300  012767  021224  155512  VDEC3:  MOV     #T020,20        ;RESTORE VECTOR
6226
6227                                  ;*****************************************************************************
6228                                  ;TEST 316       TEST THAT AN EMT CAUSES AN OVERFLOW TRAP (CHECK OF YELLOW ZONE)
6229                                  ;*****************************************************************************
6230  022306                          TS316:
6231  022306  012706  000400                  MOV     #400,%6         ;SET UP STACK TO OVERFLOW
6232  022312  012767  022330  155510          MOV     #VDEC6,30       ;SET UP INST VECTOR
6233  022320  012767  022334  155456          MOV     #VDEC5,4        ;SET UP OVERFLOW BECTOR
6234  022326  025260                          INST                    ;THIS TRAP SHOULD CAUSE OVERFLOW
6235  022330  000167  157526          VDEC6:  JMP     ERROR1          ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
6236  022334  012767  002062  155466  VDEC5:  MOV     #ERROR1,30      ;RESTORE VECTOR
6237
6238                                  ;*****************************************************************************
6239                                  ;TEST 317       TEST THAT AN TRAP CAUSES AN OVERFLOW TRAP
6240                                  ;*****************************************************************************
6241  022342                          TS317:
6242  022342  012706  000400                  MOV     #400,%6         ;SET UP STACK TO OVERFLOW
6243  022346  012767  022364  155460          MOV     #VDEC8,34       ;SET UP TRAP VECTOR
6244  022354  012767  022370  155422          MOV     #VDEC7,4        ;SET UP OVERFLOW VECTOR
6245  022362  104400                          TRAP                    ;THIS TRAP SHOULD CAUSE OVERFLOW
6246  022364  000167  157472          VDEC8:  JMP     ERROR1          ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
6247  022370  012767  021230  155436  VDEC7:  MOV     #T034,34        ;RESTORE VECTOR
6248                                  ;*****************************************************************************
6249                                  ;TEST 320       TEST THAT AN TRT CAUSES AN OVERFLOW TRAP
6250                                  ;*****************************************************************************
6251  022376                          TS320:
6252  022376  012706  000400                  MOV     #400,%6         ;SET UP STACK TO OVERFLOW
6253  022402  012767  022420  155404          MOV     #VDEC10,14      ;SET UP TRT VECTOR
6254  022410  012767  022424  155366          MOV     #VDEC9,4        ;SET UP OVERFLOW VECTOR
6255  022416  000003                          TRT                     ;THIS TRAP SHOULD CAUSE OVERFLOW
6256  022420  000167  157436          VDEC10: JMP     ERROR1          ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
6257  022424  012767  021222  155362  VDEC9:  MOV     #T014,14        ;RESTORE VECTOR
6258                                  ;*****************************************************************************
6259                                  ;TEST 321       TEST THAT AN ILLA CAUSES AN OVERFLOW TRAP
6260                                  ;*****************************************************************************
6261  022432                          TS321:
6262  022432  012706  000400                  MOV     #400,%6         ;SET UP STACK TO OVERFLOW
6263  022436  012767  022454  155340          MOV     #VDEC11,4       ;SET UP ILLA VECTOR
6264  022444  012767  022460  155332          MOV     #VDEC12,4       ;SET UP OVERFLOW VECTOR
6265  022452  004700                          ILLA                    ;THIS TRAP SHOULD CAUSE OVERFLOW
6266  022454  000167  157402          VDEC11: JMP     ERROR1          ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
```

```
6267  022460  012767  021216  155316  VDEC12: MOV     #T04,4  ;RESTORE VECTOR
6268  022466  020627  000370                  CMP     %6,#370         ;STACK PUSHED FOUR WORDS?
6269  022472  001401                          BEQ     TS322
6270  022474  104000                          EMT                     ;TRAP OVERFLOW DID NOT OCCUR
6271                                ;*****************************************************************************
6272                                ;TEST 322       TEST THAT AN ILLB CAUSES AN OVERFLOW TRAP
6273                                ;*****************************************************************************
6274  022476                        TS322:
6275  022476  012706  000400                  MOV     #400,%6         ;SET UP STACK TO OVERFLOW
6276  022502  012767  022520  155274          MOV     #VDEC13,4       ;SET UP ILLB VECTOR
6277  022510  012767  022524  155266          MOV     #VDEC14,4       ;SET UP OVERFLOW VECTOR
6278  022516  000100                          ILLB                    ;THIS TRAP SHOULD CAUSE OVERFLOW
6279  022520  000167  157336        VDEC13: JMP     ERROR1          ;USE JUMP TO GET TO ERROR BECAUSE UNSURE WHAT EMT WILL D
6280  022524  012767  021216  155252  VDEC14: MOV     #T04,4  ;RESTORE VECTOR
6281
6282                                ;*****************************************************************************
6283                                ;TEST 323       TEST FOR FALSE OVERFLOW TRAP
6284                                ;*****************************************************************************
6285  022532                        TS323:
6286
6287  022532  012767  022600  155244          MOV     #FOVER,4        ;SET UP OVERFLOW POINTER
6288  022540  012706  001002                  MOV     #1002,%6
6289  022544  005746                          TST     -(6)            ;SHOULD NOT OVERFLOW
6290  022546  012706  002002                  MOV     #2002,%6
6291  022552  005746                          TST     -(6)            ;SHOULD NOT OVERFLOW
6292  022554  012706  004002                  MOV     #4002,%6
6293  022560  005746                          TST     -(6)            ;SHOULD NOT OVERFLOW
6294  022562  012706  010002                  MOV     #10002,%6
6295  022566  005746                          TST     -(6)
6296  022570  012706  020000                  MOV     #20000,%6       ;SHOULD NOT OVERFLOW
6297  022574  005746                          TST     -(6)
6298  022576  000401                          BR      STP
6299  022600                        FOVER:
6300  022600  104000                          EMT                     ;IT OVERFLOWED,OR WRONG $TSTNM
6301  022602  012767  021216  155174  STP:    MOV     #T04,4
6302  022610  005067  155172                  CLR     6
6303                                ;*****************************************************************************
6304                                ;TEST 324       TEST THAT BIT 4 PSW WILL CAUSE A TRAP TO 14
6305                                ;*****************************************************************************
6306  022614                        TS324:
6307  022614  012706  001000                  MOV     #STBOT,SP
6308  022620  012767  022644  155166          MOV     #RETAT,RTRAP4   ;SET UP TO TRAP TO 14
6309  022626  012746  000020                  MOV     #20,-(SP)       ;PUSH T BIT
6310  022632  012746  022640                  MOV     #.+6,-(SP)      ;PUSH PC
6311  022636  000002                          RTI                     ;SET T BIT
6312  022640  000240                          NOP                     ;TRAP HERE
6313  022642  104000                          EMT                     ;TRACE BIT DID NOT TRAP!,OR WRONG $TESTN
6314  022644                        RETAT:
6315                                ;*****************************************************************************
6316                                ;TEST 325       TEST  STACK POINTER DECREMENTS
6317                                ;*****************************************************************************
6318  022644                        TS325:
6319  022644  012706  001000                  MOV     #STBOT,SP
6320  022650  012767  022674  155136          MOV     #RETBT,RTRAP4
6321  022656  012746  000020                  MOV     #20,-(SP)       ;PUSH T BIT
6322  022662  012746  022670                  MOV     #.+6,-(SP)      ;PUSH PC
```

```
6323  022666  000002                          RTI                     ;SET T BIT
6324  022670  000240                          NOP                     ;TRAP HERE
6325  022672  104000                          EMT                     ;TRACE BIT DID NOT TRAP!
6326  022674  020627  000774          RETBT:  CMP     SP,#STBOT-4
6327  022700  001401                          BEQ     TS326
6328  022702  104000                          EMT                     ;STACK POINTER WAS NOT PUSHED BY TRAP,OR WRONG $TESTN
6329                                  ;*****************************************************************************
6330                                  ;TEST 326       TEST  FOR PROPER PC ON STACK
6331                                  ;*****************************************************************************
6332  022704                          TS326:
6333  022704  012706  001000                  MOV     #STBOT,SP
6334  022710  012767  022730  155076          MOV     #RETCT,RTRAP4
6335  022716  012746  000020                  MOV     #20,-(SP)       ;PUSH T BIT
6336  022722  012746  022730                  MOV     #.+6,-(SP)      ;PUSH PC
6337  022726  000002                          RTI                     ;SET T BIT
6338                                                                  ;TRAP HERE
6339  022730  022767  022730  156036  RETCT:  CMP     #.,STBOT-4
6340  022736  001401                          BEQ     TS327
6341  022740  104000                          EMT                     ;CORRECT PC WAS NOT SAVED ON STACK,OR WRONG $TESTN
6342
6343
6344                                  ;*****************************************************************************
6345                                  ;TEST 327       TEST  THAT RTT POPS T- BIT
6346                                  ;*****************************************************************************
6347  022742                          TS327:
6348
6349  022742  012706  001000                  MOV     #STBOT,SP
6350  022746  005001                          CLR     R1              ;CLEAR R1
6351  022750  012746  000020                  MOV     #20,-(SP)
6352  022754  012746  022770                  MOV     #RTT1,-(SP)
6353  022760  012767  022776  155026          MOV     #RTT2,14
6354  022766  000006                          RTT
6355  022770  000240                  RTT1:   NOP
6356  022772  001401                          BEQ     TS330
6357  022774  104000                          EMT                     ;T-BIT DID NOT TRAP,OR WRONG $TESTN
6358
6359  022776                          RTT2:
6360                                  ;*****************************************************************************
6361                                  ;TEST 330       TEST  THAT RTT ALLOWS ONE INST. BEFORE TRAP
6362                                  ;*****************************************************************************
6363  022776                          TS330:
6364  022776  012705  177777                  MOV     #177777,%5
6365  023002  012706  001000          RTT5:   MOV     #STBOT,SP
6366  023006  012746  000020                  MOV     #20,-(SP)
6367  023012  012746  023030                  MOV     #RTT3,-(SP)
6368  023016  012767  023040  154770          MOV     #RTT4,14
6369  023024  005001                          CLR     R1              ;CLEAR R0
6370  023026  000006                          RTT                     ;SET T-BIT
6371  023030  005201                  RTT3:   INC     R1
6372  023032  005205                          INC     %5
6373  023034  001762                          BEQ     RTT5            ;DO THIS TEST NO MORE THAN 2 TIMES
6374  023036  104000                          EMT                     ;DID NOT TRAP
6375  023040  005301                  RTT4:   DEC     R1              ;SEE IF RTT ALLOWS 1 INST.
6376  023042  001403                          BEQ     RTT6
6377  023044  005205                          INC     %5              ;DO THIS TEST NO MORE THAN TWO TIMES
6378  023046  001755                          BEQ     RTT5
```

```
6379  023050  104000                          EMT                     ;RTT DID NOT ALLOW 1 INST.,OR WRONG $TESTN
6380  023052                                RTT6:
6381                                        ;*****************************************************************************
6382                                        ;TEST 331       TEST  THAT RTI DOES NOT ALLOW 1 INST.
6383                                        ;*****************************************************************************
6384  023052                                TS331:
6385  023052  012706  001000                       MOV     #STBOT,SP
6386  023056  012746  000020                       MOV     #20,-(SP)
6387  023062  012746  023100                       MOV     #RTI1,-(SP)
6388  023066  012767  023104  154720               MOV     #RTI2,14
6389  023074  005001                               CLR     R1
6390  023076  000002                               RTI                     ;SET T-BIT
6391  023100  005201                        RTI1:  INC     R1              ;RTI SHOULD NOT ALLOW THIS
6392  023102  104000                               EMT                     ;T- BIT DID NOT CAUSE TRAP
6393  023104  005701                        RTI2:  TST     R1
6394                                                                       ;RTI SHOULD NOT ALLOW 1 INST. BEFORE TRAP
6395  023106  001401                               BEQ     TS332
6396  023110  104000                               EMT                     ;RTI DID ALLOW 1 INST. BEFORE TRAP,OR WRONG $TESTN
6397
6398                                        ;*****************************************************************************
6399                                        ;TEST 332       TEST TRAP ON TRAP THAT TRACE BIT TRAPS ARE INHIBITED ON TRAP INST
6400                                        ;*****************************************************************************
6401  023112                                TS332:
6402
6403  023112  012706  001000                       MOV     #STBOT,%6
6404  023116  012767  023156  154670               MOV     #TRACE,14       ;TRACE TRAP
6405  023124  005027  000016                       CLR     #16
6406  023130  005027  000022                       CLR     #22
6407  023134  012767  023162  154656               MOV     #TONT1,20       ;IOT TRAP
6408  023142  012746  000020                       MOV     #20,-(SP)       ;PUSH T BIT
6409  023146  012746  023154                       MOV     #.+6,-(SP)      ;PUSH PC
6410  023152  000006                               RTT
6411  023154  000004                               IOT                     ;TRAP, NEW CC HAVE TRACE RESET
6412  023156                                TRACE:
6413  023156  104000                               EMT                     ;TRACE TRAP WAS NOT INHIBITED
6414  023160                                BR70:
6415  023160  104000                               EMT                     ;WRONG TSTNM,OR WRONG $TSTNM
6416  023162  012767  000016  154624       TONT1:  MOV     #16,14
6417  023170  012767  000022  154622               MOV     #22,20
```

```
6418                                    ;*****************************************************************************
6419                                    ;TEST 333       TEST THAT THE TRACE BIT IS SAVED IN THE STACK
6420                                    ;*****************************************************************************
6421  023176                            TS333:
6422  023176  012706  001000                  MOV     #STBOT,%6       ;SET UP STACK POINTER
6423  023202  012767  023226  154604          MOV     #TRC1,14        ;TRACE TRAP RETURN
6424  023210  005067  154602                  CLR     16
6425  023214  012746  000020                  MOV     #20,-(SP)       ;SET THE T BIT
6426  023220  012746  023226                  MOV     #TRC1,-(SP)
6427  023224  000002                          RTI
6428  023226  036727  155544  000020  TRC1:   BIT     STBOT-2,#20     ;CHECK FOR T BIT ON STACK
6429  023234  001001                          BNE     STP3D
6430  023236                            STP3:
6431  023236  104000                          EMT                     ;T BIT NOT SAVED ON THE STACK,OR WRONG $TSTNM
6432  023240  012767  021222  154546  STP3D:  MOV     #TO14,14
6433
```

```
                                ;*************************************************************************
                                ;THIS ROUTINE TESTS THAT NO LEGAL ADDRESS TRAPS AND THAT AN ILLEGAL
                                ;ADDRESS TRAPS TO LOCATION 4.  THIS WILL RUN ON 30K SYSTEM.  BUT IF
                                ;SWITCH REGISTER BIT 1=0, THEN THE MEMORY FROM 28K-30K IS NOT LOOKED
                                ;AT, SINCE IT MAY HAVE I/O DEVICES.  IF SWR BIT 1=1, THEN THAT AREA IS
                                ;CHECKED.  (IT SHOULD EITHER ALL TRAP OR ALL NOT TRAP).  LOC 160000
                                ;IS NO LONGER GUARANTEED TO TRAP, SINCE IT MAY CONTAIN MEMORY. LOCATION
                                ;177700 (THE UNIBUS ADDRESS FOR R0 ON OLDER SYSTEMS) IS USED FOR FORCING
                                ;A TIMEOUT IN THE EVENT THAT THERE WAS NO TIMEOUT FROM 0K-28K OR 30K.
                                ;THIS ROUTINE TESTS MEMORY UNTIL IT DOES A NXM STOP
                                ;*************************************************************************
                                ;TEST 334        TEST NON-EXISTENT ADDRESS TRAPS
                                ;*************************************************************************
023246                          TS334:
023246  005000                  1$:     CLR     R0              ;
023250  005067  154532                  CLR     6               ;
023254  012767  023340  154522          MOV     #ATRAP,4        ;SET UP ADDRESS TRAP ENTRANCE
023262  012706  001000                  MOV     #STBOT,SP       ;SET STACK POINTER
023266  105720          NOR:    TSTB    (0)+            ;IF OUTSIDE OF CORE, TRAP TO 4
023270  020027                  CMP     R0,(PC)+        ;IS POINTER INSIDE 28K (30K) CORE
023272  160000          HICORE: .WORD   160000          ;MAY BE CHANGED TO 170000 IF 30K
023274  103774                  BLO     NOR             ;TEST THE REST OF CORE
023276  012737  023312  000004          MOV     #ROTRAP,@#4     ;SET UP NEW VECTOR POINTER
023304  105737  177700                  TSTB    @#177700        ;SHOULD CAUSE A TRAP
023310                          TRPADR:
023310  104000                  EMT                     ;SHOULD HAVE TRAPED
                                ;TRAP TO HERE IF FORCING TRAP BY TESTING 177700
023312  106767  154460          ROTRAP: MFPS    STATUS
023316  005767  154454                  TST     STATUS          ;TEST PSW
023322  001401                  BEQ     1$
023324  104000                  EMT                     ;NEW PSW SHOULD HAVE BEEN ZERO
023326  026727  155442  023310  1$:     CMP     STBOT-4,#TRPADR ;TEST OLD PC AT STACK
023334  001437                  BEQ     TRAPB
023336  104000                  EMT                     ;OLD PC WAS NOT SAVED
                                ;RETURN HERE ON AN ADDRESS TRAP FROM MEMORY BELOW 28K (OR 30K)
023340  005300          ATRAP:  DEC     R0              ;
023342  010067  000032          MOV     R0,CORH         ;MOVE THE FIRST NXM LOCATION IN CORH
                                ;THIS ROUTINE DOES NXM TRAPS UNTIL IT FINDS AN EXISTENT MEMORY LOCATION
023346  013700  023272          MOV     @#HICORE,R0     ;SET UP THE HIGHEST MEM LOCATION
023352  005300                  DEC     R0              ;MAKE 1 LESS THAN THE HIGHEST CORE BOUNDARY
023354  000402                  BR      NOSUB           ;DON'T SUBTRACT 1K FIRST TIME
023356  162700  001000  CTRAP:  SUB     #1000,R0        ;SUBTRACT 1K OCTAL BYTE FROM ADDRESS
                                                        ;TO SPEED UP TESTING
023362  012767  023406  154414  NOSUB:  MOV     #BTRAP,4        ;SET UP THE VECTOR
023370  012706  001000                  MOV     #STBOT,SP
023374  005710                  TST     (R0)            ;DOES THIS MEMORY EXIST?
                                                        ;IF NXM, TRAP TO BTRAP
023376  020027          DTRAP1: CMP     R0,(PC)+        ;IF EXISTS, IS THIS THE SAME TRAP THAT CAUSED
                                                        ;TRAP TO ATRAP
023400  000000          CORH:   .WORD   0
023402  101414                  BLOS    TRAPB
023404  104000                  EMT                     ;CONTENTS OF R0 SHOULD BE LESS THAN OR EQUAL  TO CORH
                                                        ;IF THIS COMPARISON FAILS IT MEANS
                                                        ;THAT SOME LEGAL ADDRESS TRAPPED, OR
                                                        ;THAT AN ILLEGAL ADDRESS DID NOT TRAP
023406  106767  154364  BTRAP:  MFPS    STATUS
```

```
6490  023412  005767  154360          TST     STATUS
6491  023416  001401                  BEQ     1$
6492  023420  104000                  EMT                     ;NEW PSW SHOULD HAVE BEEN ZERO
6493  023422  026727  155346  023376  1$:     CMP     STBOT-4,#DTRAP1 ;CHECK IF TRAP PC IS OK
6494  023430  001752                  BEQ     CTRAP
6495  023432                          AUTO1:
6496  023432  104000                  EMT                     ;OLD PC WAS NOT SAVED OR WRONG $TESTN
6497  023434  012767  021216  154342  TRAPB:  MOV     #T04,4          ;RESET TRAP CATCHER
6498  023442  005067  154340          CLR     6               ;RESET TRAP CATCHER
6499
6500
6501                                  ;THIS ROUTINE WILL FIGURE OUT IF YOU HAVE A DL11W
6502  023446  012706  001000          MOV     #STBOT,SP       ;SET UP THE STACK POINTER
6503  023452  012767  023466  154324  MOV     #NODL,4         ;SET UP THE TRAP VECTOR
6504  023460  005767  154100          TST     TTCSR           ;TEST THE PUNCH STATUS REGISTER
6505  023464  000405                  BR      DL11W
6506  023466  012767  021216  154310  NODL:   MOV     #T04,4
6507  023474  000167  074564          JMP     SLU1ST          ;IF NO SLU FIND OUT WHY IN SLU TEST
6508  023500  012767  021216  154276  DL11W:  MOV     #T04,4
6509
6510                                  ;*****************************************************************************
6511                                  ;TEST 335       TEST THAT A TTY INTERRUPT CAUSES AN OVERFLOW TRAP
6512                                  ;*****************************************************************************
6513  023506                          TS335:
6514  023506  012767  000340  154262          MOV     #340,STATUS     ;LOCK OUT INTERRUPT
6515  023514  012706  000400                  MOV     #400,%6         ;SET UP STACK TO OVERFLOW
6516  023520  012767  023562  154256          MOV     #TDEC77,4       ;SET UP OVERFLOW TRAP
6517  023526  016767  154332  001542          MOV     64,     TEMP1   ;SAVE CONTENTS OF INTERRUPT VECTOR
6518  023534  012767  023560  154322          MOV     #TDEC8,64       ;SET UP INTERRUPT VECTOR
6519  023542  012767  000100  154014          MOV     #100,TTCSR      ;SET INTERRUPT ENABLE
6520  023550  005067  154222                  CLR     STATUS          ;ALLOW INTERRUPT TO OCCUR
6521  023554  000167  074504                  JMP     SLU1ST          ;NO INTERRUPT OCCURRED SO GO TO SLU TEST
6522                                                                  ;TO FIND OUT WHY ADD REPORT PROPER ERROR
6523  023560                          TDEC8:
6524  023560  104000                  EMT                     ;OVERFLOW TRAP DID NOT OCCUR
6525  023562  005067  153776          TDEC77: CLR     TTCSR           ;CLEAR INTERRUPT ENABLE
6526  023566  012767  021216  154210          MOV     #T04,4
6527  023574  005067  154206                  CLR     6
6528  023600  016767  001472  154256          MOV     TEMP1,  64      ;RESTORE CONTENTS OF INTERRUPT VECTOR
6529                                  ;*****************************************************************************
6530                                  ;TEST 336       TEST THAT A PENDING INTERRUPT OCCURS BEFORE TRAP
6531                                  ;*****************************************************************************
6532  023606                          TS336:
6533  023606  012706  001000          MOV     #STBOT,%6
6534  023612  012767  000340  154156          MOV     #340,STATUS     ;SET TO A HIGH PRIORITY LEVEL
6535  023620  016767  154240  001450          MOV     64,     TEMP1   ;SAVE CONTENTS OF INTERRUPT VECTOR
6536  023626  012767  023672  154230          MOV     #TR0,64
6537  023634  012767  000100  153722          MOV     #100,TTCSR      ;INTERRUPT FOR TTY PUNCH/PRINTER
6538  023642  012767  023674  154164          MOV     #BR71,34        ;TRAP VECTOR
6539  023650  012767  023676  154206          MOV     #TR2,64         ;TTY VECTOR
6540  023656  012767  000340  154152          MOV     #340,36         ;IF TRAP TRAPS, MOVE 340 TO PRIORITY
6541  023664  005067  154106                  CLR     STATUS          ;SHOULD INTERRUPT AT END OF CLR INST
6542  023670  104400                          TRAP                    ;TTY INTERRUPT SHOULD OVERRIDE TRAP
6543  023672                          TR0:
6544  023672  104000                  EMT                     ;TTY SHOULDN'T HAVE INTERRUPTED
6545  023674                          BR71:
```

```
6546  023674  104000                          EMT                     ;TRAP OCCURRED FIRST
6547  023676  005067  154134            TR2:  CLR     36
6548  023702  016767  001370  154154          MOV     TEMP1,  64      ;RESTORE CONTENTS OF INTERRUPT VECTOR
6549  023710  042767  000100  153646          BIC     #100,TTCSR
6550                                    ;*****************************************************************************
6551                                    ;TEST 337       TEST THAT A PENDING INTERRUPT, INTERRUPTS BETWEEN TRAPS
6552                                    ;*****************************************************************************
6553  023716                            TS337:
6554  023716  012706  001000                  MOV     #STBOT,%6
6555  023722  012767  000340  154046          MOV     #340,STATUS
6556  023730  012767  000100  153626          MOV     #100,TTCSR
6557  023736  012767  024004  154070          MOV     #TR3,34                 ;TRAP
6558  023744  016767  154114  001324          MOV     64,     TEMP1   ;SAVE CONTENTS OF INTERRUPT VECTOR
6559  023752  012767  024010  154104          MOV     #TR4,64                 ;TTY OUTPUT
6560  023760  012767  000340  154100          MOV     #340,66                 ;TTY OUTPUT PRIORITY
6561  023766  012767  024006  154024          MOV     #TR5,20                 ;IOT
6562  023774  012767  000340  154020          MOV     #340,22                 ;IOT PRIORITY
6563  024002  104400                          TRAP                            ;THE ACT OF TRAPPING LOWER PRIORITY
6564  024004  000004                    TR3:  IOT                             ;INTERRUPT SHOULD OCCUR IN PLACE OF IOT TRAP
6565  024006                            TR5:
6566  024006  104000                          EMT                             ;NO INTERRUPT BETWEEN TRAPS,OR WRONG $TSTNM
6567  024010  005067  154006            TR4:  CLR     22                      ;CLR IOT PRIORITY
6568  024014  005067  154046                  CLR     66
6569  024020  012767  021230  154006          MOV     #T034,34
6570  024026  016767  001244  154030          MOV     TEMP1,  64      ;RESTORE CONTENTS OF INTERRUPT VECTOR
6571  024034  012767  000022  153756          MOV     #22,20
6572  024042  042767  000100  153514          BIC     #100,TTCSR      ;CLEAR IE BIT IN SLU1 XMIT CSR
6573
6574                                    ;*****************************************************************************
6575                                    ;TEST 340       TEST THAT "RESET" GOES TO OUTSIDE WORLD
6576                                    ;*****************************************************************************
6577  024050                            TS340:
6578  024050  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
6579  024056  001403                          BEQ     70$             ;IF NO THEN DO TEST
6580  024060  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
6581  024064  001013                          BNE     TS341           ;IF NO THEN SHIP TO NEXT TEST
6582  024066                            70$:
6583  024066  016700  153470                  MOV     TKB,R0          ;MAKE SURE RECEIVER DONE IS CLEAR
6584  024072  012767  000100  153460          MOV     #100,TRCSR      ;SET INTERRUPT ENABLE
6585  024100  000005                          RESET                   ;SHOULD CLEAR INTERRUPT ENABLE
6586  024102  032767  000100  153450          BIT     #100,TRCSR      ;TEST FOR CLEAR
6587  024110  001401                          BEQ     TS341
6588  024112  104000                          EMT                     ;RESET FAILED TO CLEAR TRCSR,OR WRONG $TSTNM
6589  024114                            NODL2:
6590                                    ;*****************************************************************************
6591                                    ;TEST 341       TEST THAT RESET HAS NO EFFECT ON THE TRACE TRAP
6592                                    ;*****************************************************************************
6593  024114                            TS341:
6594  024114  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
6595  024122  001403                          BEQ     70$             ;IF NO THEN DO TEST
6596  024124  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
6597  024130  001024                          BNE     TS342           ;IF NO THEN SHIP TO NEXT TEST
6598  024132                            70$:
6599  024132  012706  001000                  MOV     #STBOT,%6       ;SET STACK
6600  024136  012767  024164  153650          MOV     #RESET2,14      ;SET UP TRACE VECTOR
6601  024144  012746  000020                  MOV     #20,-(R6)       ;SET THE T-BIT ON STACK
```

```
6602  024150  012746  024156                    MOV     #1$,-(R6)       ;MOVE NEW PC ON STACK
6603  024154  000006                            RTT
6604  024156  000005                    1$:     RESET                   ;SHOULD HAVE NO EFFECT
6605  024160  000005                            RESET                   ;NO EFFECT
6606  024162                             RESET3:
6607  024162  104000                            EMT                     ;TRACE TRAP FAILED,OR WRONG $TSTNM
6608  024164  005067  153606             RESET2: CLR     STATUS          ;CLEAR TRACK
6609  024170  005067  153622                     CLR     16              ;TRACE STATUS
6610  024174  012767  021222  153612             MOV     #T014,14
6611  024202                             SKTST2:
6612
6613                                     ;*****************************************************************************
6614                                     ;TEST 342       TEST THE 'WAIT' INSTRUCTION
6615                                     ;*****************************************************************************
6616  024202                             TS342:
6617  024202  122767  000001  154610             CMPB    #APTENV,$ENV            ;RUNING IN APT MODE?
6618  024210  001003                             BNE     1$                      ;IF NOT, DO THIS TEST
6619  024212  005767  154570                     TST     $PASS                   ;IS THIS THE FIRST PASS?
6620  024216  001051                             BNE     STP4E                   ;IF NOT FIRST PASS, SKIP TEST
6621  024220                             1$:
6622  024220  042767  000100  153336             BIC     #100,TTCSR      ;CLEAR INTERRUPT ENABLE
6623  024226  012706  001000                     MOV     #STBOT,SP       ;SET UP THE STACK
6624  024232  016767  153626  001036             MOV     64,     TEMP1   ;SAVE CONTENTS OF INTERRUPT VECTOR
6625  024240  012767  024320  153616             MOV     #WATE,64        ;SET UP THE INTERRUPT VECTOR
6626  024246  005067  153614                     CLR     66
6627  024252  105767  153306             WATE1:  TSTB    TTCSR           ;WAIT FOR READY
6628  024256  100375                             BPL     WATE1           ;TO BE UP
6629  024260  012767  000015  153300             MOV     #15,TPB         ;DO A CARRIAGE RETURN
6630  024266  105767  153272             WATE2:  TSTB    TTCSR           ;WAIT FOR READY TO COME UP
6631  024272  100375                             BPL     WATE2
6632  024274  012767  000015  153264             MOV     #15,TPB         ;DO ANOTHER CARRIAGE RETURN
6633  024302  052767  000100  153254             BIS     #100,TTCSR      ;SET THE INTERRUPT ENABLE
6634  024310  005067  153462                     CLR     STATUS          ;CLEAR THE PSW
6635  024314  000001                     WATE3:  WAIT                    ;WAIT FOR THE INTERRUPT
6636  024316  104000                             EMT                     ;WAIT INSTRUCTION DID NOT LOOP
6637  024320  005767  153452             WATE:   TST     STATUS  ;IS THE PSW CORRECT?
6638  024324  001401                             BEQ     1$
6639  024326  104000                             EMT                     ;NEW PSW SHOULD HAVE BEEN ZERO
6640  024330  026727  154440  024316     1$:     CMP     STBOT-4,#WATE3+2        ;IS THE OLD PC SAVED
6641  024336  001401                             BEQ     STP4E
6642  024340                             STP4:
6643  024340  104000                             EMT                     ;OLD PC WAS NOT SAVED OR WRONG $TESTN
6644  024342  016767  000730  153514     STP4E:  MOV     TEMP1,  64      ;RESTORE CONTENTS OF INTERRUPT VECTOR
6645  024350  042767  000100  153206             BIC     #100,TTCSR      ;CLEAR IE BIT IN SLU1 XMIT CSR
6646                                     ;*****************************************************************************
6647                                     ;TEST 343       TEST THAT USING REGISTER ADDR (177700) CAUSES TIME OUT.
6648                                     ;*****************************************************************************
6649  024356                             TS343:
6650                                             ;
6651                                             ;REGISTER ADDRESS (177700-177717) CAUSE TIME OUT WHEN USED
6652                                             ;AS PROGRAM ADDRESS BY THE CPU.
6653                                             ;
6654  024356  012706  001000                     MOV     #STBOT,SP       ;SET  STACK POINTER
6655  024362  012767  024376  153414             MOV     #RETR1,RTRAP5   ;SET TRAP RETURN ADDR
6656  024370  005237  177700             PCN1:   INC     @#177700        ;BAD ADDR REFERENCE, TRAP TO 4
6657  024374  104000                             EMT                     ;REFERENCING 177700 DID NOT CAUSE TIME OUT
```

```
6658  024376  022767  024374  154370  RETR1:  CMP     #PCN1+4,STBOT-4 ;PROPER PC STORED ON STACK?
6659  024404  001401                          BEQ     TS344
6660  024406  104000                          EMT                     ;OLD PC WAS NOT SAVED IN STACK
6661
6662                                          ;*************************************************************
6663                                          ;ODD ADDRESS USED BY A 'WORD' INSTRUCTION SHOULD NOT
6664                                          ;CAUSE A TRAP, BUT THE LOW ORDER ADDRESS BIT WOULD BE IGNORED.
6665                                  ;*****************************************************************************
6666                                  ;TEST 344       TEST ODD ADDRESS TRAP IS NOT IMPLEMENTED.
6667                                  ;*****************************************************************************
6668  024410                          TS344:
6669                                          ;
6670  024410  012737  024434  000004          MOV     #RETR2,@#RTRAP5 ;SET TRAP RETURN ADDR
6671  024416  005037  000000                  CLR     @#0             ;PUT ALL 0 IN LOC 0
6672  024422  005337  000001                  DEC     @#1             ;DECREMENT ODD ADDRESS, SHOULD NOT TRAP
6673  024426  022737  177777  000000          CMP     #-1,@#0         ;WORD LOC 0 HAS ALL ONES?
6674  024434                  RETR2:
6675  024434  001401                          BEQ     TS345
6676  024436  104000                          EMT                     ;LOC 0 DID NOT STORE -1,OR ODD ADDR REFERENCE CAUSE TRAP
6677
6678                                          ;*************************************************************
6679                                          ;
6680                                          ;USING ADDRESS 177700 IN MODE 2, CAUSES BUS ERROR, BUT
6681                                          ;THE REGISTER IN USE WILL BE INCREMENTED.
6682                                          ;
6683                                  ;*****************************************************************************
6684                                  ;TEST 345       TEST THAT IN MODE 2, BAD ADDRESS REFERENCE CAUSES BUS ERROR.
6685                                  ;*****************************************************************************
6686  024440                          TS345:
6687  024440  012737  024460  000004          MOV     #RETR3,@#RTRAP5 ;SET TRAP RETURN ADDR
6688  024446  012700  177700                  MOV     #177700,R0      ;STORES BAD MEMORY REFERENCE
6689  024452  012720  001234                  MOV     #1234,(R0)+     ;BAD ADDR REFERENCE, TRAP TO LOC 4
6690  024456  104000                          EMT                     ;ADDRESSING 177700 DID NOT CAUSE TRAP
6691  024460  022700  177702          RETR3:  CMP     #177702,R0      ;WAS R0 INCREMENTED?
6692  024464  001401                          BEQ     TS346
6693  024466  104000                          EMT                     ;R0 WAS NOT INCREMENTED
6694                                          ;*************************************************************
6695                                          ;
6696                                          ;AFTER THE FIRST BUS ERROR WAS ENCOUNTERED, AN ATTEMPT WAS MADE
6697                                          ;TO PUSH PC AND PS INTO THE STACK. HOWEVER, IF THE STACK POINTER
6698                                          ;WAS BAD, A BOUBLE BUS ERROR OCCURED.  THE STACK POINTER WOULD
6699                                          ;THEN BE SET TO LOCATION 4, OLD PC AND PS WERE PUSHED INTO
6700                                          ;LOCATIONS 0 AND 2. THE PROCESSOR WOULD TRAP TO 4 AND CONTINUE
6701                                          ;EXECUTION.
6702                                          ;
6703                                  ;*****************************************************************************
6704                                  ;TEST 346       TEST FOR DOUBLE BUS ERROR.
6705                                  ;*****************************************************************************
6706  024470                          TS346:
6707  024470  012767  024532  153306          MOV     #DBE1,RTRAP5    ;SET TRAP RETURN ADDR
6708  024476  012737  000340  000006          MOV     #340,@#6        ;SET UP PS
6709  024504  012767  024530  153276          MOV     #DBE2,RTRAP     ;SET TRAP RETURN ADDR
6710  024512  012737  000340  000012          MOV     #340,@#12       ;SET UP PS
6711  024520  012706  177700                  MOV     #177700,SP      ;SET ILLEGAL SP
6712  024524  000077                  DBE:    TRAPA                   ;ILLEGAL INSTRUCTION
6713  024526  104000                          EMT                     ;DOUBLE BUS ERROR DID NOT CAUSE TRAP
```

```
6714  024530                                DBE2:
6715  024530  104000                              EMT                     ;TRAP TO WRONG LOCATION
6716  024532  022737  024526  000000      DBE1:   CMP     #DBE+2,@#0      ;OLD PC GOT SAVED?
6717  024540  001401                              BEQ     DBE3
6718  024542  104000                              EMT                     ;OLD PC DID NOT GET SAVEDD
6719  024544  022737  000340  000002      DBE3:   CMP     #340,@#2        ;CORRECT PS SAVED?
6720  024552  001401                              BEQ     DBE4
6721  024554  104000                              EMT                     ;CORRECT PS DID NOT GET SAVE
6722  024556  022706  000000              DBE4:   CMP     #0,SP           ;SP POINTS TO LOC 0?
6723  024562  001401                              BEQ     DBE5
6724  024564  104000                              EMT                     ;SP IS NOT POINTING TO LOC 0
6725  024566  012706  001000              DBE5:   MOV     #STBOT,SP       ;RESET SP
6726  024572  012767  021216  153204              MOV     #TO4,4          ;RESET VECTOR 4
6727  024600  012767  021220  153202              MOV     #TO10,10        ;RESET VECTOR 10
6728
6729                                              ;*****************************************************************
6730                                              ;THIS TEST WILL CHECK THE SERVICE ROUTINE FOR A CONTROL CHIP ERROR.
6731                                              ;THIS IS DONE BY EXECUTING INSTRUCTIONS WHICH JUMP TO NON-EXISTENT
6732                                              ;CONTROL-CHIP. THE TEST EXECUTES AN FIS INSTRUCTION WHICH
6733                                              ;IS ILLEGAL ON ALL PROCESSORS USING THE DCF11-A CHIP SET.
6734                                              ;A CTLERR TRAPS TO LOCATION 10.
6735                                              ;THE RESET LINE IS ALSO ASSERTED FOR 1 CYCLE.
6736                                  ;*************************************************************************************
6737                                  ;TEST 347       TEST CTLERR SERVICE ROUTINE
6738                                  ;*************************************************************************************
6739  024606                          TS347:
6740
6741  024606  012706  001000                  MOV     #STBOT,R6       ;INIT STACK POINTER
6742  024612  012737  024632  000010          MOV     #1$,@#10                ;SET UP RETURN ADDR FROM TRAP
6743  024620  012737  000340  000012          MOV     #340,@#12       ;SET TRAP PRIORITY=7
6744  024626  075006                          FADD    R6              ;EXECUTE FIS INSTR..SHOULD CAUSE CTLERR
6745  024630  000000                          HALT                    ;DID NOT TRAP..CHECK CSEL LINE
6746  024632  012706  001000          1$:     MOV     #STBOT,R6       ;RE-INIT STACK POINTER
6747  024636  012767  021220  153144          MOV     #TO10,10        ;RESET VECTOR 10
6748
6749                                  ;*************************************************************************************
6750                                  ;TEST 350       TEST THAT ALL RESERVED INSTRUCTIONS TRAP
6751                                  ;*************************************************************************************
6752  024644                          TS350:
6753  024644  042767  000100  152712          BIC     #100,TTCSR
6754                                                                  ; SET UP TO SEE IF
6755  024652  013767  000010  000042          MOV     @#10,TENSAVE    ;  THIS PROCESSOR HAS THE
6756  024660  012737  024724  000010          MOV     #TRAP10,@#10    ;  FLOATING POINT OPTION
6757  024666  170127  000000                  LDFPS   #0              ;DO A FPP INSTRUCTION
6758                                                                  ;IF NO TRAP FPP INSTALLED
6759  024672  013767  025246  000356          MOV     @#FPP,FINISH    ;SO RESET END OF TABLE POINTER
6760  024700  000411                          BR      AROUND          ; THE FOLLOWING
6761
6762                                  ;* IF NO CIS OPTION TRAP TO HERE
6763  024702  042777  000040  174354  CISTRP: BIC     #40,@SWR        ;CLEAR CIS OPTION IN SWR
6764  024710  012716  024756                  MOV     #CONCIS,(SP)    ;CHANGE RETURN ADDRESS TO CONCIS LOCATION
6765  024714  000002                          RTI                     ;RETURN
6766  024716  000000                  CISADR: .WORD   0               ;DATA FOR CIS
6767  024720  000000                          .WORD   0               ;               INSTRUCTION
6768  024722  000000                  TENSAVE:.WORD   0               ; A PLACE TO STORE CONTENTS OF 10
6769  024724                          TRAP10:                         ; LEAVE THE TABLE ALONE
```

```
6770
6771  024724                                  AROUND:                         ;  CONTINUATION POINT
6772  024724  012737  000246  000244                MOV     #246,@#244        ;  RESTORE THE TRAP VECTOR
6773  024732  012737  024702  000010                MOV     #CISTRP,@#10      ;SET UP TO SEE IF THIS HAS THE CIS OPTION
6774  024740  076144                                .WORD   076144            ;EXECUTE A CMPCI INSTRUCTION
6775  024742  024716                                .WORD   CISADR            ;OPERANDS
6776  024744  024716                                .WORD   CISADR            ;          FOR CIS
6777  024746  000000                                .WORD   0                 ;                INSTRUCTION
6778  024750  052777  000040  174306                BIS     #40,@SWR          ;SET CIS PRESENT BIT
6779  024756  016737  177740  000010  CONCIS:       MOV     TENSAVE,@#10      ; RESTORE THE ILLEGAL INST. VECTOR
6780  024764  012703  025136                        MOV     #TABLE,TAB        ;TABLE POINTER
6781  024770  012305                    GIN1:       MOV     (TAB)+,FIRST      ;FIRST OR CURRENT INSTRUCTION
6782  024772  012301                                MOV     (TAB)+,LAST       ;LAST INSTRUCTION OR GROUP
6783  024774  020537  025212                        CMP     FIRST,@#CIS
6784  025000  001007                                BNE     1$
6785  025002  032777  000040  174254                BIT     #40,@SWR
6786  025010  001403                                BEQ     1$
6787  025012  012703  025246                        MOV     #FPP,TAB
6788  025016  000764                                BR      GIN1
6789  025020  020567  000232            1$:         CMP     FIRST,FINISH      ;TESTED ALL
6790  025024  001415                                BEQ     GIN3              ;YES BRANCH
6791  025026  010567  000226                        MOV     FIRST,INST        ;SET UP INST
6792  025032  005267  000222            GIN2:       INC     INST
6793  025036  012767  025072  152744                MOV     #RET,10           ;SET UP RETURN FROM TRAP
6794  025044  012706  001000                        MOV     #STBOT,SP         ;SET UP STACK POINTER
6795  025050  005067  152722                        CLR     CC                ;CLEAR PRIORITY
6796  025054  000167  000200                        JMP     INST              ;EXECUTE RESERVED INSTRUCTION
6797  025060  012767  021220  152722  GIN3:       MOV     #TO10,10          ;RESET VECTOR 10
6798  025066  000167  000252                        JMP     THRPRT            ;JUMP TO EIS TEST
6799
6800                                    ;TRAPPING SHOULD SEND YOU HERE
6801  025072  020627  000774            RET:        CMP     SP,#STBOT-4       ;TEST DECREMENT OF SP
6802  025076  001401                                BEQ     RET1
6803  025100  104000                                EMT                       ;WRONG DECREMENT
6804  025102  026727  153666  025262    RET1:       CMP     STBOT-4,#INST+2   ;LOC OF INST UNINCREMENTED
6805  025110  001401                                BEQ     RET2
6806  025112  104000                                EMT                       ;INST INC ON TRAP
6807  025114  005767  153656            RET2:       TST     STBOT-2
6808  025120  001401                                BEQ     RET3
6809  025122                            RET4:
6810  025122  104000                                EMT                       ;CONDITION CODES SET ON TRAP OR WRONG $TSTNM
6811  025124  026701  000130            RET3:       CMP     INST,LAST
6812  025130  001717                                BEQ     GIN1              ;SET UP NEW GROUP
6813  025132  000167  177674                        JMP     GIN2              ;FINISH OLD GROUP
6814                                                                          ;END OF INSTRUCTION GROUP
6815  025136  000007                    TABLE:      7                         ;END OF OPERATE
6816  025140  000077                                77
6817  025142  000207                                207                       ;RTS,RT1,JMP
6818  025144  000227                                227
6819  025146  006777                                6777
6820  025150  007777                                7777
6821  025152  075037                                075037
6822  025154  076017                                76017
6823  025156  076032                                76032
6824  025160  076037                                76037
6825  025162  076045                                76045
```

```
6826  025164  076047                    76047
6827  025166  076077                    76077
6828  025170  076127                    76127
6829  025172  076132                    76132
6830  025174  076137                    76137
6831  025176  076145                    76145
6832  025200  076147                    76147
6833  025202  076157                    76157
6834  025204  076167                    76167
6835  025206  076177                    76177
6836  025210  076777                    76777
6837  025212  076017            CIS:    76017
6838  025214  076032                    76032
6839  025216  076037                    76037
6840  025220  076045                    76045
6841  025222  076047                    76047
6842  025224  076077                    76077
6843  025226  076127                    76127
6844  025230  076132                    76132
6845  025232  076137                    76137
6846  025234  076145                    76145
6847  025236  076147                    76147
6848  025240  076157                    76157
6849  025242  076167                    76167
6850  025244  076177                    76177
6851  025246  167777            FPP:    167777          ;  START OF THE FPP INSTRUCTIONS
6852  025250  177700                    177700
6853  025252  177716                    177716
6854  025254  177777                    177777
6855  025256  025256            FINISH: .               ;END FLAG
6856  025260  000000            INST:   HALT            ;WILL CONTINUE RESERVED INST
6857  025262  000000                    HALT            ;SHOULD TRAP TO LOC 10
6858  025264  000000                    HALT            ;LOC 10 SHOULD SEND YOU TO
6859  025266  000000                    HALT            ;RET
6860  025270  000000                    HALT
6861                              .SBTTL ** STARTING OF EIS TEST **
6862
6863
6864          000000            DUMMY=  0
6865          000051            F=      51
6866          000176            N=      176
6867
6868  025272                    COUNT:
6869          025274                    .=COUNT+2
6870  025274                    PSWORD:
6871          025276                    .=PSWORD+2
6872  025276                    TEMP1:
6873          025300                    .=TEMP1+2
6874  025300                    TEMP2:
6875          025302                    .=TEMP2+2
6876  025302                    TEMP3:
6877          025304                    .=TEMP3+2
6878  025304                    TEMP4:
6879          025306                    .=TEMP4+2
6880  025306  000000            TEMP5:  .WORD
6881  025310  000000            TEMP6:  .WORD
```

```
 6882  025312  177771
 6883  025314  025312
 6884  025316  177772
 6885  025320  177777
 6886  025322  040000
 6887  025324  025322
 6888  025326  040000
 6889  025330  177776
 6890  025332  000002
 6891  025334  025332
 6892  025336  000002
 6893  025340  177566
 6894  025342  177564
 6895
 6896
 6897
 6898
 6899
 6900
```

```
S1:     -7
S2:     S1
S3:     -6
S4:     -1
S5:     40000
S6:     S5
S7:     40000
S8:     -2
S9:     2
S10:    S9
S11:    2
$TPB:   177566
$TPS:   177564
```

```
6901  025344                          THRPRT:
6902
6903  025344  012705  001004                  MOV     #$TESTN,R5      ;MAKE R5 POINT TO WHERE TEST # IS SAVED
6904  025350  005037  025272                  CLR     @#COUNT         ;CLEAR THE COUNTER
6905  025354  012715  000001                  MOV     #1,(R5)         ;INITIALIZE TEST NUMBER
6906  025360  012706  001000                  MOV     #STBOT,SP       ;** STACK AT STBOT **
6907  025364  012737  000001  025276  2$:     MOV     #1,@#TEMP1      ;TEMP1=1
6908  025372  005037  025300                  CLR     @#TEMP2         ;TEMP2=0
6909  025376  012737  000001  025302          MOV     #1,@#TEMP3      ;TEMP3=1
6910  025404  005037  025304                  CLR     @#TEMP4         ;TEMP4=0
6911  025410  106427  000000                  MTPS    #0
6912
6913
```

```
6914
6915                                    ;*********************************
6916                                    ;     ASH INSTRUCTION TESTS
6917                                    ;*********************************
6918
6919
6920
6921                                        ;*************
6922                                        ; TESTS 1-36
6923                                        ;*************
6924
6925
6926
6927  025414  013700  025276        ASTART: MOV     @#TEMP1,%0      ;LOAD R0 WITH THE CONTENTS OF TEMP1
6928  025420  032737  000001  001006        BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
6929  025426  001004                        BNE     2$              ;IF NOT THEN GO TO 2$
6930  025430  013701  025300                MOV     @#TEMP2,R1      ;OTHERWISE EXECUTE THE INSTRUCTION
6931                                                                ;IN MODE 0 USING R1
6932  025434  072001                        ASH     R1,R0
6933  025436  000402                        BR      4$
6934  025440  072067  177634        2$:     ASH     TEMP2,%0        ;SHIFT R0 BY THE NUMBER SPECIFIED BY TEMP2
6935  025444  106737  025274        4$:     MFPS    @#PSWORD        ;SAVE PS
6936  025450  123737  025304  025274        CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
6937  025456  001401                        BEQ     11$
6938  025460  104000                        EMT                     ;THE PS IS NOT EQUAL TO 0
6939  025462  005237  025272        11$:    INC     @#COUNT         ;INCREMENT THE COUNTER
6940  025466  023700  025302                CMP     @#TEMP3,%0      ;IS THE RESULT IN R0 EQUAL TO TEMP3?
6941  025472  001401                        BEQ     12$
6942  025474                        6$:
6943  025474  104000                        EMT                     ;EITHER INCORRECT R0 OR INCORRECT SEQUENCE
6944  025476  021537  025272        12$:    CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE
6945                                                                ;COUNTER?
6946  025502  001374                        BNE     6$              ;IF NOT GO TO THE HLT ABOVE
6947  025504  005215                        INC     (R5)
6948  025506  021527  000037                CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT
6949                                                                ;BY 14. AND RIGHT BY 14.?
6950  025512  002011                        BGE     8$
6951  025514  005237  025300                INC     @#TEMP2
6952  025520  006367  177556                ASL     TEMP3           ;SHIFT TEMP3 LEFT.
6953  025524  021527  000020                CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
6954  025530  001004                        BNE     REGR1
6955  025532  000167  000670                JMP     NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
6956  025536  004767  000712        8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
6957  025542  013701  025276        REGR1:  MOV     @#TEMP1,%1      ;LOAD R1 WITH THE CONTENTS OF TEMP1
6958  025546  032737  000001  001006        BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
6959  025554  001004                        BNE     2$              ;IF NOT THEN GO TO 2$
6960  025556  013702  025300                MOV     @#TEMP2,R2      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
6961  025562  072102                        ASH     R2,R1           ;USING R1
6962  025564  000402                        BR      4$
6963  025566  072167  177506        2$:     ASH     TEMP2,%1        ;SHIFT R1 BY THE NUMBER SPECIFIED BY TEMP2
6964  025572  106737  025274        4$:     MFPS    @#PSWORD        ;SAVE PS
6965  025576  123737  025304  025274        CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
6966  025604  001401                        BEQ     11$
6967  025606  104000                        EMT                     ;THE PS IS NOT EQUAL TO 0
6968  025610  005237  025272        11$:    INC     @#COUNT         ;INCREMENT THE COUNTER
6969  025614  023701  025302                CMP     @#TEMP3,%1      ;IS THE RESULT IN R1 EQUAL TO TEMP3?
```

```
6970  025620  001401                          BEQ     12$
6971  025622                          6$:
6972  025622  104000                          EMT                     ;EITHER INCORRECT R1 OR INCORRECT SEQUENCE
6973  025624  021537  025272          12$:    CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
6974  025630  001374                          BNE     6$              ;IF NOT GO TO THE HLT ABOVE
6975  025632  005215                          INC     (R5)
6976  025634  021527  000037                  CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT
6977                                                                  ;BY 14. AND RIGHT BY 14.?
6978  025640  002011                          BGE     8$
6979  025642  005237  025300                  INC     @#TEMP2
6980  025646  006367  177430                  ASL     TEMP3           ;SHIFT TEMP3 LEFT
6981  025652  021527  000020                  CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
6982  025656  001004                          BNE     REGR2
6983  025660  000167  000542                  JMP     NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
6984  025664  004767  000564          8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
6985  025670  013702  025276          REGR2:  MOV     @#TEMP1,%2      ;LOAD R2 WITH THE CONTENTS OF TEMP1
6986  025674  032737  000001  001006          BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
6987  025702  001004                          BNE     2$              ;IF NOT THEN GO TO 2$
6988  025704  013703  025300                  MOV     @#TEMP2,R3      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
6989  025710  072203                          ASH     R3,R2           ;USING R2
6990  025712  000402                          BR      4$
6991  025714  072267  177360          2$:     ASH     TEMP2,%2        ;SHIFT R2 BY THE NUMBER SPECIFIED BY TEMP2
6992  025720  106737  025274          4$:     MFPS    @#PSWORD        ;SAVE PS
6993  025724  123737  025304  025274          CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
6994  025732  001401                          BEQ     11$
6995  025734  104000                          EMT                     ;THE PS IS NOT EQUAL TO 0
6996  025736  005237  025272          11$:    INC     @#COUNT
6997  025742  023702  025302                  CMP     @#TEMP3,%2      ;IS THE RESULT IN R2 EQUAL TO TEMP3?
6998  025746  001401                          BEQ     12$
6999  025750                          6$:
7000  025750  104000                          EMT                     ;EITHER INCORRECT R2 OR INCORRECT SEQUENCE
7001  025752  021537  025272          12$:    CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
7002  025756  001374                          BNE     6$              ;IF NOT GO TO THE HLT ABOVE
7003  025760  005215                          INC     (R5)
7004  025762  021527  000037                  CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED
7005                                                                  ;LEFT BY 14, AND RIGHT BY 14.?
7006  025766  002011                          BGE     8$
7007  025770  005237  025300                  INC     @#TEMP2
7008  025774  006367  177302                  ASL     TEMP3           ;SHIFTED TEMP3 LEFT
7009  026000  021527  000020                  CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
7010  026004  001004                          BNE     REGR3
7011  026006  000167  000414                  JMP     NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
7012  026012  004767  000436          8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
7013  026016  013703  025276          REGR3:  MOV     @#TEMP1,%3      ;LOAD R3 WITH THE CONTENTS OF TEMP1
7014  026022  032737  000001  001006          BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
7015  026030  001004                          BNE     2$              ;IF NOT THEN GO TO 2$
7016  026032  013704  025300                  MOV     @#TEMP2,R4      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
7017  026036  072304                          ASH     R4,R3           ;USING R3
7018  026040  000402                          BR      4$
7019  026042  072367  177232          2$:     ASH     TEMP2,%3        ;SHIFT R3 BY THE NUMBER SPECIFIED BY TEMP2
7020  026046  106737  025274          4$:     MFPS    @#PSWORD        ;SAVE PS
7021  026052  123737  025304  025274          CMPB    @#TEMP4,@#PSWORD;IS THE PS = TEMP4 ?
7022  026060  001401                          BEQ     11$
7023  026062  104000                          EMT                     ;THE PS IS NOT EQUAL TO 0.
7024  026064  005237  025272          11$:    INC     @#COUNT
7025  026070  023703  025302                  CMP     @#TEMP3,%3      ;IS THE RESULT IN R3 EQUAL TO TEMP3?
```

```
7026  026074  001401                          BEQ     12$
7027  026076                          6$:
7028  026076  104000                          EMT                     ;EITHER INCORRECT R3 OR INCORRECT SEQUENCE
7029  026100  021537  025272          12$:    CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
7030  026104  001374                          BNE     6$              ;IF NOT GO TO THE HLT ABOVE
7031  026106  005215                          INC     (R5)
7032  026110  021527  000037                  CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED
7033                                                                  ;LEFT BY 14, AND RIGHT BY 14.?
7034  026114  002010                          BGE     8$
7035  026116  005237  025300                  INC     @#TEMP2
7036  026122  006367  177154                  ASL     TEMP3           ;SHIFT TEMP3 LEFT?
7037  026126  021527  000020                  CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
7038  026132  001003                          BNE     REGR4
7039  026134  000534                          BR      NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
7040  026136  004767  000312          8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
7041  026142  013704  025276          REGR4:  MOV     @#TEMP1,%4      ;LOAD R4 WITH THE CONTENTS OF TEMP1
7042  026146  010501                          MOV     R5,R1           ;SAVE R5
7043  026150  032737  000001  001006          BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
7044  026156  001004                          BNE     2$              ;IF NOT THEN GO TO 2$
7045  026160  013705  025300                  MOV     @#TEMP2,R5      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
7046  026164  072405                          ASH     R5,R4           ;USING R4
7047  026166  000402                          BR      4$
7048  026170  072467  177104          2$:     ASH     TEMP2,%4        ;SHIFT R4 BY THE NUMBER SPECIFIED BY TEMP2
7049  026174  106737  025274          4$:     MFPS    @#PSWORD        ;SAVE PS
7050  026200  123737  025304  025274          CMPB    @#TEMP4,@#PSWORD;IS PS = TEMP4 ?
7051  026206  001401                          BEQ     11$
7052  026210  104000                          EMT                     ;THE PS IS NOT EQUAL TO 0
7053  026212  005237  025272          11$:    INC     @#COUNT
7054  026216  023704  025302                  CMP     @#TEMP3,%4      ;IS THE RESULT IN R4 EQUAL TO TEMP3?
7055  026222  001401                          BEQ     12$
7056  026224                          6$:
7057  026224  104000                          EMT                     ;EITHER INCORRECT R4 OR INCORRECT SEQUENCE
7058  026226  010105                  12$:    MOV     R1,R5           ;RESTORE R5
7059  026230  021537  025272                  CMP     (R5),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
7060  026234  001373                          BNE     6$              ;IF NOT GO TO THE HLT ABOVE
7061  026236  005215                          INC     (R5)
7062  026240  021527  000037                  CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN
7063                                                                  ;SHIFTED LEFT BY 14. AND RIGHT BY 14.?
7064  026244  002010                          BGE     8$
7065  026246  005237  025300                  INC     @#TEMP2
7066  026252  006367  177024                  ASL     TEMP3           ;SHIFT TEMP3 LEFT
7067  026256  021527  000020                  CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTER BEEN SHIFTED BY 14.?
7068  026262  001003                          BNE     REGR5
7069  026264  000460                          BR      NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
7070  026266  004767  000162          8$:     JSR     PC,TST37        ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
7071  026272  010501                  REGR5:  MOV     R5,R1           ;SAVE R5
7072  026274  013705  025276                  MOV     @#TEMP1,%5      ;LOAD R5 WITH THE CONTENTS OF TEMP1
7073  026300  032737  000001  001006          BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
7074  026306  001004                          BNE     2$              ;IF NOT THEN GO TO 2$
7075  026310  013700  025300                  MOV     @#TEMP2,R0      ;OTHERWISE EXECUTE ASH INSTRUCTION IN MODE 0
7076  026314  072500                          ASH     R0,R5           ;USING R5
7077  026316  000402                          BR      4$
7078  026320  072567  176754          2$:     ASH     TEMP2,%5        ;SHIFT R5 BY THE NUMBER SPECIFIED BY TEMP2
7079  026324  106737  025274          4$:     MFPS    @#PSWORD        ;SAVE PS
7080  026330  123737  025304  025274          CMPB    @#TEMP4,@#PSWORD;IS PS = TEMP4 ?
7081  026336  001401                          BEQ     11$
```

```
7082  026340  104000                        EMT                     ;THE PS IS NOT EQUAL TO 0.
7083  026342  005237  025272        11$:    INC     @#COUNT
7084  026346  023705  025302                CMP     @#TEMP3,%5      ;IS THE RESULT IN R5 EQUAL TO TEMP3?
7085  026352  001401                        BEQ     12$
7086  026354                        6$:
7087  026354  104000                        EMT                     ;EITHER INCORRECT R5 OR INCORRECT SEQUENCE
7088  026356  021137  025272        12$:    CMP     (R1),@#COUNT    ;IS THE TEST NUMBER EQUAL TO THE COUNTER?
7089  026362  001374                        BNE     6$              ;IF NOT GO TO THE HLT ABOVE
7090  026364  010105                        MOV     R1,R5           ;RESTORE R5
7091  026366  005215                        INC     (R5)
7092  026370  021527  000037                CMP     (R5),#37        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED
7093                                                                ;LEFT BY 14. AND RIGHT BY 14.?
7094  026374  002010                        BGE     8$              ;IF SO GO AND CONTINUE THE REST OF THE PROGRAM
7095  026376  005237  025300                INC     @#TEMP2
7096  026402  006367  176674                ASL     TEMP3           ;SHIFT TEMP3 LEFT
7097  026406  021527  000020                CMP     (R5),#20        ;HAS THE CONTENTS OF REGISTERS BEEN SHIFTED LEFT BY 14.?
7098  026412  001405                        BEQ     NEGAT           ;IF SO GO TO NEGAT AND INITIATE RIGHT SHIFT
7099  026414  000402                        BR      10$
7100  026416  004767  000032        8$:     JSR     PC,TST37
7101  026422  000167  176766        10$:    JMP     ASTART          ;GO BACK TO START
7102  026426  012737  040000  025276 NEGAT:  MOV     #40000,@#TEMP1  ;TEMP1=40000
7103  026434  012737  177762  025300        MOV     #177762,@#TEMP2 ;TEMP2=177762
7104  026442  012737  000001  025302        MOV     #1,@#TEMP3      ;TEMP3=1
7105  026450  000167  176740                JMP     ASTART
7106  026454  021527  000037        TST37:  CMP     (R5),#37        ;IS IT TEST 37?
7107  026460  001013                        BNE     TST40           ;IF NOT THEN TRY TEST 40
7108  026462  005037  025276                CLR     @#TEMP1         ;0
7109  026466  012737  000020  025300        MOV     #16.,@#TEMP2    ;SHIFTED BY 16
7110  026474  005037  025302                CLR     @#TEMP3         ;IS=0
7111  026500  012737  000004  025304        MOV     #4,@#TEMP4      ;AND PS=4
7112  026506  000207                        RTS     PC
7113  026510  021527  000040        TST40:  CMP     (R5),#40        ;IS IT TEST 40?
7114  026514  001003                        BNE     TST41           ;IF NOT THEN TRY TEST 41
7115  026516  005037  025300                CLR     @#TEMP2         ;0 SHIFTED BY 0=0 AND PS=4
7116  026522  000207                        RTS     PC
7117  026524  021527  000041        TST41:  CMP     (R5),#41        ;IS IT TEST 41?
7118  026530  001004                        BNE     TST42           ;IF NOT THEN TRY TEST 42
7119  026532  012737  177760  025300        MOV     #-16.,@#TEMP2   ;0 SHIFTED BY -16.=0 AND PS=4
7120  026540  000207                        RTS     PC
7121  026542  021527  000042        TST42:  CMP     (R5),#42        ;IS IT TEST 42?
7122  026546  001013                        BNE     TST43           ;IF NOT THEN TRY TEST 43
7123  026550  012737  100000  025276        MOV     #100000,@#TEMP1 ;100000
7124  026556  005237  025300                INC     @#TEMP2         ;SHIFTED BY -15
7125  026562  005337  025302                DEC     @#TEMP3         ;IS=-1
7126  026566  012737  000010  025304        MOV     #10,@#TEMP4             ;AND PS=10
7127  026574  000207                        RTS     PC
7128  026576  021527  000043        TST43:  CMP     (R5),#43        ;IS IT TEST 43?
7129  026602  001012                        BNE     TST44           ;IF NOT THEN IF NOT THEN TRY TEST 44
7130  026604  012737  125252  025276        MOV     #125252,@#TEMP1 ;125252
7131  026612  012737  177777  025300        MOV     #-1,@#TEMP2     ;SHIFTED BY -1
7132  026620  012737  152525  025302        MOV     #152525,@#TEMP3 ;IS=152525 AND PS=10
7133  026626  000207                        RTS     PC
7134  026630  021527  000044        TST44:  CMP     (R5),#44        ;IS IT TEST 44?
7135  026634  001012                        BNE     TST45           ;IF NOT THEN TRY TEST 45
7136  026636  012737  000001  025300        MOV     #1,@#TEMP2      ;125252 SHIFTED BY 1
7137  026644  012737  052524  025302        MOV     #52524,@#TEMP3  ;IS=52524
```

```
7138  026652  012737  000003  025304          MOV     #3,@#TEMP4      ;AND PS=3
7139  026660  000207                          RTS     PC
7140  026662  021527  000045          TST45:  CMP     (R5),#45        ;IS IT TEST 45?
7141  026666  001012                          BNE     TST46           ;IF NOT THEN TRY TEST 46
7142  026670  012737  177776  025300          MOV     #-2,@#TEMP2     ;125252 SHIFTED BY -2
7143  026676  012737  165252  025302          MOV     #165252,@#TEMP3 ;IS=165252
7144  026704  012737  000011  025304          MOV     #11,@#TEMP4     ;AND PS=11
7145  026712  000207                          RTS     PC
7146  026714  021527  000046          TST46:  CMP     (R5),#46        ;IS IT TEST 46?
7147  026720  001014                          BNE     TST47           ;IF NOT THEN TRY TEST 47
7148  026722  012737  177777  025276          MOV     #-1,@#TEMP1     ;-1
7149  026730  012737  000020  025300          MOV     #16.,@#TEMP2    ;SHIFTED BY 15.
7150  026736  005037  025302                  CLR     @#TEMP3         ;IS=0
7151  026742  012737  000007  025304          MOV     #7,@#TEMP4      ;AND PS=7
7152  026750  000207                          RTS     PC
7153  026752  021527  000047          TST47:  CMP     (R5),#47        ;IS IT TEST 47?
7154  026756  001011                          BNE     TST50           ;IF NOT THEN TRY TEST 50
7155  026760  005337  025300                  DEC     @#TEMP2         ;-1 SHIFTED BY 15
7156  026764  012737  100000  025302          MOV     #100000,@#TEMP3 ;IS=100000
7157  026772  012737  000011  025304          MOV     #11,@#TEMP4     ;AND PS=11
7158  027000  000207                          RTS     PC
7159  027002  021527  000050          TST50:  CMP     (R5),#50        ;IS IT TEST 50
7160  027006  001007                          BNE     ENT51           ;IF NOT THEN TRY TEST 51
7161  027010  012737  137777  025276          MOV     #137777,@#TEMP1 ;137777 SHIFTED BY 15. IS=100000
7162  027016  012737  000013  025304          MOV     #13,@#TEMP4     ;AND PS=13
7163  027024  000207                          RTS     PC
7164  027026  021527  000051          ENT51:  CMP     (R5),#51        ;IS IT ENTERING TEST 51?
7165  027032  001401                          BEQ     1$
7166  027034  104000                          EMT                     ;TEST NUMBER GOOFED
7167
7168  027036  005726                  1$:     TST     (SP)+           ;RESTORE STACK POINTER
7169  027040  012704  177771                  MOV     #-7,%4
7170  027044  012702  025312                  MOV     #S1,%2
7171  027050  012703  025314                  MOV     #S2,%3
7172                                  ;*****************************************************************************
7173                                  ;TEST:51   11/34   ASH     125252 SHIFTED BY #5 = 52500  PS = 3
7174                                  ;*****************************************************************************
7175
7176  027054  012701  125252          TST51:  MOV     #125252,%1      ;LOAD R1 WITH 125252
7177  027060  072127  000005                  ASH     #5,%1           ;SHIFT R1 BY #5
7178  027064  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7179  027070  122737  000003  025274          CMPB    #3,@#PSWORD     ;IS THE PS 3?
7180  027076  001401                          BEQ     11$
7181  027100  104000                          EMT                     ;THE PS IS NOT EQUAL TO 3
7182  027102  022701  052500          11$:    CMP     #52500,%1       ;IS THE RESULT 52500?
7183  027106  001401                          BEQ     12$
7184  027110  104000                          EMT                     ;R1 IS NOT EQUAL TO 52500 OR INCORRECT SEQUENCE
7185  027112  005215                  12$:    INC     (R5)
7186
7187
7188
7189                                  ;*****************************************************************************
7190                                  ;TEST:52   11/34   ASH     125252 SHIFTED BY @S2 = 177525  PS = 10
7191                                  ;*****************************************************************************
7192
7193  027114  012700  125252          TST52:  MOV     #125252,%0      ;LOAD R0 WITH 125252
```

```
7194  027120  072077  176170            ASH     @S2,%0          ;SHIFT R0 BY @S2
7195  027124  106737  025274            MFPS    @#PSWORD        ;SAVE PS
7196  027130  122737  000010  025274    CMPB    #10,@#PSWORD    ;IS THE PS 10?
7197  027136  001401                    BEQ     11$
7198  027140  104000                    EMT                     ;THE PS IS NOT EQUAL TO 10
7199  027142  022700  177525    11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7200  027146  001401                    BEQ     12$
7201  027150  104000                    EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7202  027152  005215    12$:    INC     (R5)
7203
7204
7205
7206                    ;*************************************************************************
7207                    ;TEST:53   11/34   ASH     125252 SHIFTED BY @#S1 = 177525  PS = 10
7208                    ;*************************************************************************
7209
7210  027154  012700  125252    TST53:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7211  027160  072037  025312            ASH     @#S1,%0         ;SHIFT R0 BY @#S1
7212  027164  106737  025274            MFPS    @#PSWORD        ;SAVE PS
7213  027170  122737  000010  025274    CMPB    #10,@#PSWORD    ;IS THE PS 10?
7214  027176  001401                    BEQ     11$
7215  027200  104000                    EMT                     ;THE PS IS NOT EQUAL TO 10
7216  027202  022700  177525    11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7217  027206  001401                    BEQ     12$
7218  027210  104000                    EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7219  027212  005215    12$:    INC     (R5)
7220
7221
7222
7223                    ;*************************************************************************
7224                    ;TEST:54   11/34   ASH     125252 SHIFTED BY (2) = 177525  PS = 10
7225                    ;*************************************************************************
7226
7227  027214  012700  125252    TST54:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7228  027220  072012            ASH     (2),%0          ;SHIFT R0 BY (2)
7229  027222  106737  025274            MFPS    @#PSWORD        ;SAVE PS
7230  027226  122737  000010  025274    CMPB    #10,@#PSWORD    ;IS THE PS 10?
7231  027234  001401                    BEQ     11$
7232  027236  104000                    EMT                     ;THE PS IS NOT EQUAL TO 10
7233  027240  022700  177525    11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7234  027244  001401                    BEQ     12$
7235  027246  104000                    EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7236  027250  005215    12$:    INC     (R5)
7237
7238
7239
7240                    ;*************************************************************************
7241                    ;TEST:55   11/34   ASH     125252 SHIFTED BY (2)+ = 177525  PS = 10
7242                    ;*************************************************************************
7243
7244  027252  012700  125252    TST55:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7245  027256  072022            ASH     (2)+,%0         ;SHIFT R0 BY (2)+
7246  027260  106737  025274            MFPS    @#PSWORD        ;SAVE PS
7247  027264  122737  000010  025274    CMPB    #10,@#PSWORD    ;IS THE PS 10?
7248  027272  001401                    BEQ     11$
7249  027274  104000                    EMT                     ;THE PS IS NOT EQUAL TO 10
```

```
7250  027276  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7251  027302  001401                          BEQ     12$
7252  027304  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7253  027306  005215                  12$:    INC     (R5)
7254
7255
7256
7257                                  ;*************************************************************************
7258                                  ;TEST:56   11/34   ASH     125252 SHIFTED BY -(2) = 177525  PS = 10
7259                                  ;*************************************************************************
7260
7261  027310  012700  125252          TST56:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7262  027314  072042                          ASH     -(2),%0         ;SHIFT R0 BY -(2)
7263  027316  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7264  027322  122737  000010  025274        CMPB    #10,@#PSWORD    ;IS THE PS 10?
7265  027330  001401                          BEQ     11$
7266  027332  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
7267  027334  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7268  027340  001401                          BEQ     12$
7269  027342  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7270  027344  005215                  12$:    INC     (R5)
7271
7272
7273
7274                                  ;*************************************************************************
7275                                  ;TEST:57   11/34   ASH     125252 SHIFTED BY 2(3) = 177252  PS = 11
7276                                  ;*************************************************************************
7277
7278  027346  012700  125252          TST57:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7279  027352  072063  000002                  ASH     2(3),%0         ;SHIFT R0 BY 2(3)
7280  027356  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7281  027362  122737  000011  025274        CMPB    #11,@#PSWORD    ;IS THE PS 11?
7282  027370  001401                          BEQ     11$
7283  027372  104000                          EMT                     ;THE PS IS NOT EQUAL TO 11
7284  027374  022700  177252          11$:    CMP     #177252,%0      ;IS THE RESULT 177252?
7285  027400  001401                          BEQ     12$
7286  027402  104000                          EMT                     ;R0 IS NOT EQUAL TO 177252 OR INCORRECT SEQUENCE
7287  027404  005215                  12$:    INC     (R5)
7288
7289
7290
7291                                  ;*************************************************************************
7292                                  ;TEST:60   11/34   ASH     125252 SHIFTED BY @(3) = 177525  PS = 10
7293                                  ;*************************************************************************
7294
7295  027406  012700  125252          TST60:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7296  027412  072073  000000                  ASH     @(3),%0         ;SHIFT R0 BY @(3)
7297  027416  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7298  027422  122737  000010  025274        CMPB    #10,@#PSWORD    ;IS THE PS 10?
7299  027430  001401                          BEQ     11$
7300  027432  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
7301  027434  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7302  027440  001401                          BEQ     12$
7303  027442  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7304  027444  005215                  12$:    INC     (R5)
7305
```

```
7306
7307
7308                                    ;*************************************************************************
7309                                    ;TEST:61   11/34   ASH     125252 SHIFTED BY @(3)+ = 177525  PS = 10
7310                                    ;*************************************************************************
7311
7312  027446  012700  125252            TST61:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7313  027452  072033                            ASH     @(3)+,%0        ;SHIFT R0 BY @(3)+
7314  027454  106737  025274                    MFPS    @#PSWORD        ;SAVE PS
7315  027460  122737  000010  025274            CMPB    #10,@#PSWORD    ;IS THE PS 10?
7316  027466  001401                            BEQ     11$
7317  027470  104000                            EMT                     ;THE PS IS NOT EQUAL TO 10
7318  027472  022700  177525            11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7319  027476  001401                            BEQ     12$
7320  027500  104000                            EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7321  027502  005215                    12$:    INC     (R5)
7322
7323
7324                                    ;*************************************************************************
7325                                    ;TEST:62   11/34   ASH     125252 SHIFTED BY @-(3) = 177525  PS = 10
7326                                    ;*************************************************************************
7327
7328
7329  027504  012700  125252            TST62:  MOV     #125252,%0      ;LOAD R0 WITH 125252
7330  027510  072053                            ASH     @-(3),%0        ;SHIFT R0 BY @-(3)
7331  027512  106737  025274                    MFPS    @#PSWORD        ;SAVE PS
7332  027516  122737  000010  025274            CMPB    #10,@#PSWORD    ;IS THE PS 10?
7333  027524  001401                            BEQ     11$
7334  027526  104000                            EMT                     ;THE PS IS NOT EQUAL TO 10
7335  027530  022700  177525            11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7336  027534  001401                            BEQ     12$
7337  027536  104000                            EMT                     ;R0 IS NOT EQUAL TO 177525 OR INCORRECT SEQUENCE
7338  027540  005215                    12$:    INC     (R5)
7339
7340
7341
```

```
                                ;**********************************
                                ;     ASHC INSTRUCTION TESTS
                                ;**********************************



                                        ;*************
                                        ;TESTS 63-157
                                        ;*************

027542 012737 000062 025272             MOV     #62,@#COUNT
027550 005037 025276                    CLR     @#TEMP1         ;TEMP1=0
027554 012737 000001 025300             MOV     #1,@#TEMP2      ;TEMP2=1
027562 005037 025302                    CLR     @#TEMP3         ;TEMP3=0
027566 005037 025304                    CLR     @#TEMP4         ;TEMP4=0
027572 012737 000001 025306             MOV     #1,@#TEMP5      ;TEMP5=1
027600 005037 025310                    CLR     @#TEMP6         ;0 1 SHIFTED BY 0=0 1, PS=0

027604 010502                   REG01:  MOV     R5,R2           ;SAVE R5
027606 013700 025276                    MOV     @#TEMP1,%0      ;PLACE THE CONTENTS OF TEMP1 IN REGISTER 0
027612 013701 025300                    MOV     @#TEMP2,%0!1    ;PLACE THE CONTENTS OF TEMP2 IN REGISTER 1
027616 000241                           CLC
027620 032737 000001 001006             BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
027626 001004                           BNE     2$              ;IF NOT THEN GO TO 2$
027630 013705 025302                    MOV     @#TEMP3,R5      ;OTHERWISE EXECUTE ASHC INSTRUCTION IN MODE 0
027634 073005                           ASHC    R5,R0           ;USING R0
027636 000402                           BR      4$
027640 073067 175436            2$:     ASHC    TEMP3,%0        ;ASHC REGISTER 0 BY THE CONTENTS OF TEMP3
027644 106737 025274            4$:     MFPS    @#PSWORD        ;SAVE PS
027650 123737 025310 025274             CMPB    @#TEMP6,@#PSWORD;COMPARE PS WITH THE CONTENTS OF TEMP6
027656 001401                           BEQ     11$
027660 104000                           EMT                     ;WRONG PS
027662 005237 025272            11$:    INC     @#COUNT
027666 023700 025304                    CMP     @#TEMP4,%0      ;IS THE RESULT IN R0 SAME AS TEMP4?
027672 001401                           BEQ     12$
027674 104000                           EMT                     ;WRONG RESULT IN R0
027676 023701 025306            12$:    CMP     @#TEMP5,%1      ;IS THE RESULT IN R1 SAME AS TEMP5?
                                                                ;TEMP1 TEMP2 SHIFTED BY TEMP3=TEMP4 TEMP5
                                                                ;AND PS=TEMP6
027702 001401                           BEQ     13$
027704 104000                           EMT                     ;WRONG RESULT IN R1
027706 010205                   13$:    MOV     R2,R5           ;RESTORE R5
027710 021537 025272                    CMP     (R5),@#COUNT    ;IS TEST NUMBER=COUNTER?
027714 001401                           BEQ     14$
027716 104000                           EMT                     ;NO
027720 005215                   14$:    INC     (R5)
027722 021527 000160                    CMP     (R5),#160       ;HAVE THE FIRST 159 TEST BEEN EXECUTED?
027726 002014                           BGE     6$              ;YES
027730 005237 025302                    INC     @#TEMP3
027734 000241                           CLC
027736 006137 025306                    ROL     @#TEMP5         ;ROTATE TEMP5 LEFT BY 1 PLACE
027742 006137 025304                    ROL     @#TEMP4         ;INTRODUCE CARRY FROM TEMP4 IN TEMP5
027746 021527 000121                    CMP     (R5),#121       ;IS IT TEST 121?
027752 001004                           BNE     REGR23
```

```
7398  027754  004467  000344            JSR     R4,RITSH        ;IF SO THEN GO AND INITIATE RIGHT SHIFT
7399  027760  004767  000374     6$:    JSR     %7,TST160
7400  027764  013702  025276     REGR23: MOV    @#TEMP1,%2      ;PLACE THE CONTENTS OF TEMP1 IN REGISTER 2
7401  027770  013703  025300             MOV     @#TEMP2,%2!1    ;PLACE THE CONTENTS OF TEMP2 IN REGISTER 3
7402  027774  000241                     CLC
7403  027776  032737  000001  001006     BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
7404  030004  001004                     BNE     2$              ;IF NOT THEN GO TO 2$
7405  030006  013704  025302             MOV     @#TEMP3,R4      ;OTHERWISE EXECUTE ASHC INSTRUCTION IN MODE 0
7406  030012  073204                     ASHC    R4,R2           ;USING R2
7407  030014  000402                     BR      4$
7408  030016  073267  175260     2$:    ASHC    TEMP3,%2        ;ASHC REGISTER 2 BY THE CONTETNS OF TEMP3
7409  030022  106737  025274     4$:    MFPS    @#PSWORD        ;SAVE PS
7410  030026  123737  025310  025274     CMPB    @#TEMP6,@#PSWORD;COMPARE PS WITH THE CONTENTS OF TEMP6
7411  030034  001401                     BEQ     11$
7412  030036  104000                     EMT                     ;WRONG PS
7413  030040  005237  025272     11$:   INC     @#COUNT
7414  030044  023702  025304             CMP     @#TEMP4,%2      ;IS THE RESULT IN R2 SAME AS TEMP4?
7415  030050  001401                     BEQ     12$
7416  030052  104000                     EMT                     ;WRONG RESULT IN R2
7417  030054  023703  025306     12$:   CMP     @#TEMP5,%3      ;IS THE RESULT IN R3 SAME AS TEMP5?
7418                                             ;TEMP1 TEMP2 SHIFTED BY TEMP3=TEMP4 TEMP5
7419                                             ;AND PS=TEMP6
7420  030060  001401                     BEQ     13$
7421  030062  104000                     EMT                     ;WRONG RESULT IN R1
7422  030064  021537  025272     13$:   CMP     (R5),@#COUNT    ;IS TEST NUMBER=COUNTER?
7423  030070  001401                     BEQ     14$
7424  030072  104000                     EMT                     ;NO
7425  030074  005215     14$:           INC     (R5)
7426  030076  021527  000160             CMP     (R5),#160       ;HAVE THE FIRST 159 TEST BEEN EXECUTED?
7427  030102  002014                     BGE     6$              ;YES
7428  030104  005237  025302             INC     @#TEMP3
7429  030110  000241                     CLC
7430  030112  006137  025306             ROL     @#TEMP5         ;ROTATE TEMP5 LEFT BY 1 PLACE
7431  030116  006137  025304             ROL     @#TEMP4         ;INTRODUCE CARRY FROM TEMP5 IN TEMP4
7432  030122  021527  000121             CMP     (R5),#121       ;IS IT TEST 121?
7433  030126  001004                     BNE     REG45
7434  030130  004467  000170             JSR     R4,RITSH        ;IF SO THEN GO AND INITIATE RIGHT SHIFT
7435  030134  004767  000220     6$:    JSR     %7,TST160
7436  030140  010501     REG45:         MOV     R5,R1           ;SAVE R5
7437  030142  013704  025276             MOV     @#TEMP1,%4      ;PLACE THE CONTENTS OF TEMP1 IN REGISTER 4
7438  030146  013705  025300             MOV     @#TEMP2,%4!1    ;PLACE THE CONTENTS OF TEMP2 IN REGISTER 5
7439  030152  000241                     CLC
7440  030154  032737  000001  001006     BIT     #1,@#$PASS      ;IS IT AN EVEN PASS ?
7441  030162  001004                     BNE     2$              ;IF NOT THEN GO TO 2$
7442  030164  013700  025302             MOV     @#TEMP3,R0      ;OTHERWISE EXECUTE ASHC INSTRUCTION IN MODE 0
7443  030170  073400                     ASHC    R0,R4           ;USING R4
7444  030172  000402                     BR      4$
7445  030174  073467  175102     2$:    ASHC    TEMP3,%4        ;ASHC REGISTER 4 BY THE CONTENTS OF TEMP3
7446  030200  106737  025274     4$:    MFPS    @#PSWORD        ;SAVE PS
7447  030204  123737  025310  025274     CMPB    @#TEMP6,@#PSWORD;COMPARE PS WITH THE CONTENTS OF TEMP6
7448  030212  001401                     BEQ     11$
7449  030214  104000                     EMT                     ;WRONG PS
7450  030216  005237  025272     11$:   INC     @#COUNT
7451  030222  023704  025304             CMP     @#TEMP4,%4      ;IS THE RESULT IN R4 SAME AS TEMP4?
7452  030226  001401                     BEQ     12$
7453  030230  104000                     EMT                     ;WRONG RESULT IN R4
```

```
7454  030232  023705  025306          12$:    CMP     @#TEMP5,%5      ;IS THE RESULT IN R5 SAME AS TEMP5?
7455                                                                    ;TEMP1 TEMP2 SHIFTED BY TEMP3=TEMP4 TEMP5
7456                                                                    ;AND PS=TEMP6
7457  030236  001401                          BEQ     13$
7458  030240  104000                          EMT                     ;WRONG RESULT IN R5
7459  030242  021137  025272          13$:    CMP     (R1),@#COUNT    ;IS TEST NUMBER=COUNTER?
7460  030246  001401                          BEQ     14$
7461  030250  104000                          EMT                     ;NO
7462  030252  010105          14$:    MOV     R1,R5                   ;RESTORE R5
7463  030254  005215                          INC     (R5)
7464  030256  021527  000160                  CMP     (R5),#160       ;HAVE THE FIRST 159 TEST BEEN EXECUTED?
7465  030262  002014                          BGE     6$              ;YES
7466  030264  005237  025302                  INC     @#TEMP3
7467  030270  000241                          CLC
7468  030272  006137  025306                  ROL     @#TEMP5         ;ROTATE TEMP5 LEFT BY 1 PLACE
7469  030276  006137  025304                  ROL     @#TEMP4         ;INTRODUCE CARRY FROM TEMP5 IN TEMP4
7470  030302  021527  000121                  CMP     (R5),#121       ;IS IT TEST 121?
7471  030306  001004                          BNE     8$
7472  030310  004467  000010                  JSR     R4,RITSH        ;IF SO THEN GO AND INITIATE RIGHT SHIFT
7473  030314  004767  000040          6$:     JSR     %7,TST160
7474  030320  000167  177260          8$:     JMP     REG01
7475  030324  022424                  RITSH:  CMP     (R4)+,(R4)+     ;MAKE R4 POINT TO THE NEXT REG TAG
7476  030326  012737  040000  025276          MOV     #40000,@#TEMP1  ;TEMP1=4000
7477  030334  005037  025300                  CLR     @#TEMP2         ;TEMP2=0
7478  030340  012737  177742  025302          MOV     #-30.,@#TEMP3   ;TEMP3=-30
7479  030346  005037  025304                  CLR     @#TEMP4         ;TEMP4=0
7480  030352  005237  025306                  INC     @#TEMP5         ;TEMP5=1
7481  030356  000204                          RTS     R4
7482  030360  021527  000160          TST160: CMP     (R5),#160       ;IS IT TEST 160
7483  030364  001010                          BNE     TST161          ;IF NOT THEN TRY TEST 161
7484  030366  005037  025276                  CLR     @#TEMP1         ;0 0 SHIFTED BY 0
7485  030372  005037  025304                  CLR     @#TEMP4         ;IS EQUAL TO 0 0
7486  030376  012737  000004  025310          MOV     #4,@#TEMP6      ;AND PS=4
7487  030404  000207                          RTS     %7
7488  030406  021527  000161          TST161: CMP     (R5),#161       ;IS IT TEST 161
7489  030412  001004                          BNE     TST162
7490  030414  012737  177746  025302          MOV     #-32,@#TEMP3    ;0 0 SHIFTED BY -32=0 0, PS=4
7491  030422  000207                          RTS     %7
7492  030424  021527  000162          TST162: CMP     (R5),#162       ;IS IT TEST 162
7493  030430  001004                          BNE     TST163          ;IF NOT THEN TRY TEST 163
7494  030432  012737  000032  025302          MOV     #32,@#TEMP3     ;0 0 SHIFTED BY 32=0 0, PS=4
7495  030440  000207                          RTS     %7
7496  030442  021527  000163          TST163: CMP     (R5),#163       ;IS IT TEST 163?
7497  030446  001016                          BNE     TST164          ;IF NOT THEN TRY TEST 164
7498  030450  012737  052525  025276          MOV     #52525,@#TEMP1  ;52525 0
7499  030456  012737  177760  025302          MOV     #-16.,@#TEMP3   ;SHIFTED BY -16.
7500  030464  005037  025304                  CLR     @#TEMP4
7501  030470  012737  052525  025306          MOV     #52525,@#TEMP5  ;IS EQUAL TO 0 52525
7502  030476  005037  025310                  CLR     @#TEMP6         ;AND PS = 0
7503  030502  000207                          RTS     %7
7504  030504  021527  000164          TST164: CMP     (R5),#164       ;IS IT TEST 164?
7505  030510  001014                          BNE     TST165          ;IF NOT THEN TRY TEST 165
7506  030512  012737  125252  025276          MOV     #125252,@#TEMP1 ;125252 0 SHIFTED BY -16.
7507  030520  005337  025304                  DEC     @#TEMP4
7508  030524  012737  125252  025306          MOV     #125252,@#TEMP5 ;IS EQUAL TO -1 125252
7509  030532  012737  000010  025310          MOV     #10,@#TEMP6     ;AND PS=10
```

```
7510  030540  000207                        RTS     %7
7511  030542  021527  000165        TST165: CMP     (R5),#165       ;IS IT TEST 165?
7512  030546  001007                        BNE     TST166          ;IF NOT THEN TRY TEST 166
7513  030550  012737  177777  025276        MOV     #-1,@#TEMP1     ;-1 0 SHIFTED BY -16
7514  030556  012737  177777  025306        MOV     #-1,@#TEMP5     ;IS EQUAL TO -1 -1, AND PS=10
7515  030564  000207                        RTS     %7
7516  030566  021527  000166        TST166: CMP     (R5),#166       ;IS IT TEST 166?
7517  030572  001011                        BNE     TST167          ;IF NOT THEN TRY TEST 167
7518  030574  012737  100000  025276        MOV     #100000,@#TEMP1 ;100000 0
7519  030602  012737  177740  025302        MOV     #-32.,@#TEMP3   ;SHIFTED BY -32 IS EQUAL TO -1 -1
7520  030610  005237  025310                INC     @#TEMP6         ;AND PS=11
7521  030614  000207                        RTS     %7
7522  030616  021527  000167        TST167: CMP     (R5),#167       ;IS IT TEST 167?
7523  030622  001014                        BNE     TST170          ;IF NOT THEN TRY TEST 170
```

```
 7524  030624  005037  025276          CLR     @#TEMP1
 7525  030630  005337  025300          DEC     @#TEMP2         ;0 -1
```

```
7526  030634  012737  000020  025302          MOV     #16.,@#TEMP3    ;SHIFTED BY 16.
7527  030642  005037  025306                  CLR     @#TEMP5         ;IS EQUAL TO -1 0
7528  030646  005237  025310                  INC     @#TEMP6         ;AND PS=12
7529  030652  000207                          RTS     %7
7530  030654  021527  000170          TST170: CMP     (R5),#170       ;IS IT TEST 170?
```

```
7531  030660  001007                BNE     TST171          ;IF NOT THEN TRY TEST 171
```

```
7532  030662  012737  125252  025300          MOV     #125252,@#TEMP2 ;0 125252 SHIFTED BY 16
7533  030670  012737  125252  025304          MOV     #125252,@#TEMP4 ;IS EQUAL TO 125252 0, AND PS=12
7534  030676  000207                          RTS     %7
7535  030700  021527  000171          TST171: CMP     (R5),#171       ;IS IT TEST 171?
7536  030704  001010                          BNE     TST172          ;IF NOT THEN TRY TEST 172
7537  030706  005337  025302                  DEC     @#TEMP3         ;0 125252 SHIFTED BY 15
7538  030712  012737  052525  025304          MOV     #52525,@#TEMP4  ;IS EQUAL TO 52525 0
7539  030720  005037  025310                  CLR     @#TEMP6         ;AND PS=0
7540  030724  000207                          RTS     %7
7541  030726  021527  000172          TST172: CMP     (R5),#172       ;IS IT TEST 172?
7542  030732  001006                          BNE     TST173          ;IF NOT THEN TRY TEST 173
7543  030734  012737  052525  025300          MOV     #52525,@#TEMP2  ;0 52525
7544  030742  005237  025302                  INC     @#TEMP3         ;SHIFTED BY 16. IS EQUAL TO 52525 0, AND PS=0
7545  030746  000207                          RTS     %7
7546  030750  021527  000173          TST173: CMP     (R5),#173       ;IS IT TEST 173?
7547  030754  001014                          BNE     TST174          ;IF NOT THEN TRY TEST 174
7548  030756  012737  177777  025300          MOV     #-1,@#TEMP2     ;0 -1
7549  030764  005337  025302                  DEC     @#TEMP3         ;SHIFTED BY 15.
7550  030770  012737  077777  025304          MOV     #77777,@#TEMP4
7551  030776  012737  100000  025306          MOV     #100000,@#TEMP5 ;IS EQUAL TO 77777 100000, AND PS=0
7552  031004  000207                          RTS     %7
7553  031006  021527  000174          TST174: CMP     (R5),#174       ;IS IT TEST 174?
7554  031012  001013                          BNE     TST175          ;IF NOT THEN TRY TEST 175
7555  031014  012737  100000  025276          MOV     #100000,@#TEMP1
7556  031022  005337  025300                  DEC     @#TEMP2         ;100000 -2 SHIFTED BY 15.
7557  031026  005037  025306                  CLR     @#TEMP5         ;IS EQUAL TO 77777 0
7558  031032  012737  000002  025310          MOV     #2,@#TEMP6      ;AND PS=2
7559  031040  000207                          RTS     %7
7560  031042  021527  000175          TST175: CMP     (R5),#175       ;IS IT TEST 175?
7561  031046  001015                          BNE     ENT176          ;IF NOT THEN TRY TEST 176
7562  031050  012737  177777  025276          MOV     #-1,@#TEMP1
7563  031056  005037  025300                  CLR     @#TEMP2         ;-1 0
7564  031062  005237  025302                  INC     @#TEMP3         ;SHIFTED BY 16.
7565  031066  005037  025304                  CLR     @#TEMP4         ;IS EQUAL TO 0 0
7566  031072  012737  000007  025310          MOV     #7,@#TEMP6      ;AND PS=7
```

```
7567  031100  000207                        RTS     %7
7568  031102  021527  000176        ENT176: CMP     (R5),#176       ;IS THE PROGRM ENTERING TEST 176?
7569  031106  001401                        BEQ     1$
7570  031110  104000                        EMT                     ;TEST NUMBER GOOFED
7571
7572  031112  005726                1$:     TST     (SP)+           ;RESTORE STACK POINTER
7573
7574                                ;*****************************************************************
7575                                ;TEST:176       1 SHIFTED BY 8. = 400  PS = 0
7576                                ;*****************************************************************
7577
7578  031114                        TST176:
7579  031114  012701  000000                MOV     #DUMMY,%1       ;LOAD R1 WITH DUMMY
7580  031120  012701  000001                MOV     #1,%1!1         ;LOAD R1!1 WITH 1
7581  031124  000241                        CLC
7582  031126  073127  000010                ASHC    #8.,%1          ;SHIFT R1,R1!1 BY 8.
7583  031132  106737  025274                MFPS    @#PSWORD        ;SAVE PS
7584  031136  122737  000000  025274        CMPB    #0,@#PSWORD     ;IS THE PS 0?
7585  031144  001401                        BEQ     11$
7586  031146  104000                        EMT                     ;THE PS IS NOT EQUAL TO 0
7587  031150  022701  000400        11$:    CMP     #400,%1         ;IS THE RESULT 400?
7588  031154  001401                        BEQ     13$
7589  031156  104000                        EMT                     ;R1 IS NOT EQUAL TO 400
7590  031160                        13$:
7591  031160  005215                        INC     (R5)
7592
7593
7594                                ;*****************************************************************
7595                                ;TEST:177       -1 SHIFTED BY 15. = 100000  PS = 11
7596                                ;*****************************************************************
7597
7598  031162                        TST177:
7599  031162  012703  000000                MOV     #DUMMY,%3       ;LOAD R3 WITH DUMMY
7600  031166  012703  177777                MOV     #-1,%3!1        ;LOAD R3!1 WITH -1
7601  031172  000241                        CLC
7602  031174  073327  000017                ASHC    #15.,%3         ;SHIFT R3,R3!1 BY 15.
7603  031200  106737  025274                MFPS    @#PSWORD        ;SAVE PS
7604  031204  122737  000011  025274        CMPB    #11,@#PSWORD    ;IS THE PS 11?
7605  031212  001401                        BEQ     11$
7606  031214  104000                        EMT                     ;THE PS IS NOT EQUAL TO 11
7607  031216  022703  100000        11$:    CMP     #100000,%3      ;IS THE RESULT 100000?
7608  031222  001401                        BEQ     13$
7609  031224  104000                        EMT                     ;R3 IS NOT EQUAL TO 100000
7610  031226                        13$:
7611  031226  005215                        INC     (R5)
7612
7613
7614                                ;*****************************************************************
7615                                ;TEST:200       52525 SHIFTED BY 0 = 52525  PS = 0
7616                                ;*****************************************************************
7617
7618  031230                        TST200:
7619  031230  010501                        MOV     R5,R1           ;SAVE R5
7620  031232  012705  000000                MOV     #DUMMY,%5       ;LOAD R5 WITH DUMMY
7621  031236  012705  052525                MOV     #52525,%5!1     ;LOAD R5!1 WITH 52525
7622  031242  000241                        CLC
```

```
7623  031244  073527  000000          ASHC    #0,%5           ;SHIFT R5,R5!1 BY 0
7624  031250  106737  025274          MFPS    @#PSWORD        ;SAVE PS
7625  031254  122737  000000  025274  CMPB    #0,@#PSWORD     ;IS THE PS 0?
7626  031262  001401                  BEQ     11$
7627  031264  104000                  EMT                     ;THE PS IS NOT EQUAL TO 0
7628  031266  022705  052525  11$:    CMP     #52525,%5       ;IS THE RESULT 52525?
7629  031272  001401                  BEQ     13$
7630  031274  104000                  EMT                     ;R5 IS NOT EQUAL TO 52525
7631  031276                  13$:
7632  031276  010105                  MOV     R1,R5           ;RESTORE R5
7633  031300  005215                  INC     (R5)
7634
7635
7636                          ;*****************************************************************
7637                          ;TEST:201       20010 SHIFTED BY -13. = 101  PS = 0
7638                          ;*****************************************************************
7639
7640  031302                  TST201:
7641  031302  012701  000000          MOV     #DUMMY,%1       ;LOAD R1 WITH DUMMY
7642  031306  012701  020010          MOV     #20010,%1!1     ;LOAD R1!1 WITH 20010
7643  031312  000241                  CLC
7644  031314  073127  177763          ASHC    #-13.,%1        ;SHIFT R1,R1!1 BY -13.
7645  031320  106737  025274          MFPS    @#PSWORD        ;SAVE PS
7646  031324  122737  000000  025274  CMPB    #0,@#PSWORD     ;IS THE PS 0?
7647  031332  001401                  BEQ     11$
7648  031334  104000                  EMT                     ;THE PS IS NOT EQUAL TO 0
7649  031336  022701  000101  11$:    CMP     #101,%1         ;IS THE RESULT 101?
7650  031342  001401                  BEQ     13$
7651  031344  104000                  EMT                     ;R1 IS NOT EQUAL TO 101
7652  031346                  13$:
7653  031346  005215                  INC     (R5)
7654
7655
7656                          ;*****************************************************************
7657                          ;TEST:202       -1 SHIFTED BY 16. = 0  PS = 11
7658                          ;*****************************************************************
7659
7660  031350                  TST202:
7661  031350  012703  000000          MOV     #DUMMY,%3       ;LOAD R3 WITH DUMMY
7662  031354  012703  177777          MOV     #-1,%3!1        ;LOAD R3!1 WITH -1
7663  031360  000241                  CLC
7664  031362  073327  000020          ASHC    #16.,%3         ;SHIFT R3,R3!1 BY 16.
7665  031366  106737  025274          MFPS    @#PSWORD        ;SAVE PS
7666  031372  122737  000011  025274  CMPB    #11,@#PSWORD    ;IS THE PS 11?
7667  031400  001401                  BEQ     11$
7668  031402  104000                  EMT                     ;THE PS IS NOT EQUAL TO 11
7669  031404  022703  000000  11$:    CMP     #0,%3           ;IS THE RESULT 0?
7670  031410  001401                  BEQ     13$
7671  031412  104000                  EMT                     ;R3 IS NOT EQUAL TO 0
7672  031414                  13$:
7673  031414  005215                  INC     (R5)
7674
7675
7676                          ;*****************************************************************
7677                          ;TEST:203       1 SHIFTED BY -1 = 100000  PS = 1
7678                          ;*****************************************************************
```

```
                                        TST203:
031416                                          MOV     R5,R1           ;SAVE R5
031416  010501                                  MOV     #DUMMY,%5       ;LOAD R5 WITH DUMMY
031420  012705  000000                          MOV     #1,%5!1         ;LOAD R5!1 WITH 1
031424  012705  000001                          CLC
031430  000241                                  ASHC    #-1,%5          ;SHIFT R5,R5!1 BY -1
031432  073527  177777                          MFPS    @#PSWORD        ;SAVE PS
031436  106737  025274                          CMPB    #1,@#PSWORD     ;IS THE PS 1?
031442  122737  000001  025274                  BEQ     11$
031450  001401                                  EMT                     ;THE PS IS NOT EQUAL TO 1
031452  104000                          11$:    CMP     #100000,%5      ;IS THE RESULT 100000?
031454  022705  100000                          BEQ     13$
031460  001401                                  EMT                     ;R5 IS NOT EQUAL TO 100000
031462  104000                          13$:
031464                                          MOV     R1,R5           ;RESTORE R5
031464  010105                                  INC     (R5)
031466  005215
                                        ;******************************************************************************
                                        ;TEST:204       125252 SHIFTED BY -16. = 125252  PS = 11
                                        ;******************************************************************************
                                        TST204:
031470                                          MOV     #DUMMY,%1       ;LOAD R1 WITH DUMMY
031470  012701  000000                          MOV     #125252,%1!1    ;LOAD R1!1 WITH 125252
031474  012701  125252                          CLC
031500  000241                                  ASHC    #-16.,%1        ;SHIFT R1,R1!1 BY -16.
031502  073127  177760                          MFPS    @#PSWORD        ;SAVE PS
031506  106737  025274                          CMPB    #11,@#PSWORD    ;IS THE PS 11?
031512  122737  000011  025274                  BEQ     11$
031520  001401                                  EMT                     ;THE PS IS NOT EQUAL TO 11
031522  104000                          11$:    CMP     #125252,%1      ;IS THE RESULT 125252?
031524  022701  125252                          BEQ     13$
031530  001401                                  EMT                     ;R1 IS NOT EQUAL TO 125252
031532  104000                          13$:
031534                                          INC     (R5)
031534  005215
                                        ;******************************************************************************
                                        ;TEST:205       125252 125252 SHIFTED BY 21. = 52500 000000  PS = 3
                                        ;******************************************************************************
                                        TST205:
031536                                          MOV     #125252,%2      ;LOAD R2 WITH 125252
031536  012702  125252                          MOV     #125252,%2!1    ;LOAD R2!1 WITH 125252
031542  012703  125252                          CLC
031546  000241                                  ASHC    #21.,%2         ;SHIFT R2,R2!1 BY 21.
031550  073227  000025                          MFPS    @#PSWORD        ;SAVE PS
031554  106737  025274                          CMPB    #3,@#PSWORD     ;IS THE PS 3?
031560  122737  000003  025274                  BEQ     11$
031566  001401                                  EMT                     ;THE PS IS NOT EQUAL TO 3
031570  104000                          11$:    CMP     #52500,%2       ;IS THE RESULT 52500?
031572  022702  052500                          BEQ     12$
031576  001401                                  EMT                     ;R2 IS NOT EQUAL TO 52500
031600  104000                          12$:    CMP     #000000,%2!1    ;IS THE RESULT 000000?
031602  022703  000000
```

```
7735  031606  001401                        BEQ     13$
7736  031610  104000                        EMT                     ;R2!1 IS NOT EQUAL TO 000000
7737  031612                          13$:
7738  031612  005215                        INC     (R5)
7739
7740
7741
7742  031614  012702  177771                MOV     #-7,%2
7743  031620  012703  025312                MOV     #S1,%3
7744  031624  012704  025314                MOV     #S2,%4
7745
7746                                  ;*****************************************************************************
7747                                  ;TEST:206       125252 125252 SHIFTED BY S1 = 177525 52525  PS = 10
7748                                  ;*****************************************************************************
7749
7750  031630                          TST206:
7751  031630  012700  125252                MOV     #125252,%0              ;LOAD R0 WITH 125252
7752  031634  012701  125252                MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
7753  031640  000241                        CLC
7754  031642  073067  173444                ASHC    S1,%0           ;SHIFT R0,R0!1 BY S1
7755  031646  106737  025274                MFPS    @#PSWORD        ;SAVE PS
7756  031652  122737  000010  025274        CMPB    #10,@#PSWORD    ;IS THE PS 10?
7757  031660  001401                        BEQ     11$
7758  031662  104000                        EMT                     ;THE PS IS NOT EQUAL TO 10
7759  031664  022700  177525          11$:  CMP     #177525,%0      ;IS THE RESULT 177525?
7760  031670  001401                        BEQ     12$
7761  031672  104000                        EMT                     ;R0 IS NOT EQUAL TO 177525
7762  031674  022701  052525          12$:  CMP     #52525,%0!1     ;IS THE RESULT 52525?
7763  031700  001401                        BEQ     13$
7764  031702  104000                        EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
7765  031704                          13$:
7766  031704  005215                        INC     (R5)
7767
7768
7769                                  ;*****************************************************************************
7770                                  ;TEST:207       125252 125252 SHIFTED BY @S2 = 177525 52525  PS = 10
7771                                  ;*****************************************************************************
7772
7773  031706                          TST207:
7774  031706  012700  125252                MOV     #125252,%0              ;LOAD R0 WITH 125252
7775  031712  012701  125252                MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
7776  031716  000241                        CLC
7777  031720  073077  173370                ASHC    @S2,%0          ;SHIFT R0,R0!1 BY @S2
7778  031724  106737  025274                MFPS    @#PSWORD        ;SAVE PS
7779  031730  122737  000010  025274        CMPB    #10,@#PSWORD    ;IS THE PS 10?
7780  031736  001401                        BEQ     11$
7781  031740  104000                        EMT                     ;THE PS IS NOT EQUAL TO 10
7782  031742  022700  177525          11$:  CMP     #177525,%0      ;IS THE RESULT 177525?
7783  031746  001401                        BEQ     12$
7784  031750  104000                        EMT                     ;R0 IS NOT EQUAL TO 177525
7785  031752  022701  052525          12$:  CMP     #52525,%0!1     ;IS THE RESULT 52525?
7786  031756  001401                        BEQ     13$
7787  031760  104000                        EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
7788  031762                          13$:
7789  031762  005215                        INC     (R5)
7790
```

```
                        ;*****************************************************************************
                        ;TEST:210       125252 125252 SHIFTED BY @#S1 = 177525 52525  PS = 10
                        ;*****************************************************************************

                        TST210:
                                MOV     #125252,%0              ;LOAD R0 WITH 125252
                                MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
                                CLC
                                ASHC    @#S1,%0         ;SHIFT R0,R0!1 BY @#S1
                                MFPS    @#PSWORD        ;SAVE PS
                                CMPB    #10,@#PSWORD    ;IS THE PS 10?
                                BEQ     11$
                                EMT                     ;THE PS IS NOT EQUAL TO 10
                        11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
                                BEQ     12$
                                EMT                     ;R0 IS NOT EQUAL TO 177525
                        12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
                                BEQ     13$
                                EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
                        13$:
                                INC     (R5)

                        ;*****************************************************************************
                        ;TEST:211       125252 125252 SHIFTED BY (3) = 177525 52525  PS = 10
                        ;*****************************************************************************

                        TST211:
                                MOV     #125252,%0              ;LOAD R0 WITH 125252
                                MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
                                CLC
                                ASHC    (3),%0          ;SHIFT R0,R0!1 BY (3)
                                MFPS    @#PSWORD        ;SAVE PS
                                CMPB    #10,@#PSWORD    ;IS THE PS 10?
                                BEQ     11$
                                EMT                     ;THE PS IS NOT EQUAL TO 10
                        11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
                                BEQ     12$
                                EMT                     ;R0 IS NOT EQUAL TO 177525
                        12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
                                BEQ     13$
                                EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
                        13$:
                                INC     (R5)

                        ;*****************************************************************************
                        ;TEST:212       125252 125252 SHIFTED BY (3)+ = 177525 52525  PS = 10
                        ;*****************************************************************************

                        TST212:
                                MOV     #125252,%0              ;LOAD R0 WITH 125252
                                MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
                                CLC
                                ASHC    (3)+,%0         ;SHIFT R0,R0!1 BY (3)+
```

```
7847  032132  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7848  032136  122737  000010  025274          CMPB    #10,@#PSWORD    ;IS THE PS 10?
7849  032144  001401                          BEQ     11$
7850  032146  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
7851  032150  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7852  032154  001401                          BEQ     12$
7853  032156  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
7854  032160  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
7855  032164  001401                          BEQ     13$
7856  032166  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
7857  032170                          13$:
7858  032170  005215                          INC     (R5)
7859
7860
7861                                  ;*************************************************************************
7862                                  ;TEST:213       125252 125252 SHIFTED BY -(3) = 177525 52525  PS = 10
7863                                  ;*************************************************************************
7864
7865  032172                          TST213:
7866  032172  012700  125252                  MOV     #125252,%0              ;LOAD R0 WITH 125252
7867  032176  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
7868  032202  000241                          CLC
7869  032204  073043                          ASHC    -(3),%0         ;SHIFT R0,R0!1 BY -(3)
7870  032206  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7871  032212  122737  000010  025274          CMPB    #10,@#PSWORD    ;IS THE PS 10?
7872  032220  001401                          BEQ     11$
7873  032222  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
7874  032224  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7875  032230  001401                          BEQ     12$
7876  032232  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
7877  032234  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
7878  032240  001401                          BEQ     13$
7879  032242  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
7880  032244                          13$:
7881  032244  005215                          INC     (R5)
7882
7883
7884                                  ;*************************************************************************
7885                                  ;TEST:214       125252 125252 SHIFTED BY 2(4) = 177252 125252  PS = 11
7886                                  ;*************************************************************************
7887
7888  032246                          TST214:
7889  032246  012700  125252                  MOV     #125252,%0              ;LOAD R0 WITH 125252
7890  032252  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
7891  032256  000241                          CLC
7892  032260  073064  000002                  ASHC    2(4),%0         ;SHIFT R0,R0!1 BY 2(4)
7893  032264  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7894  032270  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS THE PS 11?
7895  032276  001401                          BEQ     11$
7896  032300  104000                          EMT                     ;THE PS IS NOT EQUAL TO 11
7897  032302  022700  177252          11$:    CMP     #177252,%0      ;IS THE RESULT 177252?
7898  032306  001401                          BEQ     12$
7899  032310  104000                          EMT                     ;R0 IS NOT EQUAL TO 177252
7900  032312  022701  125252          12$:    CMP     #125252,%0!1    ;IS THE RESULT 125252?
7901  032316  001401                          BEQ     13$
7902  032320  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 125252 OR INCORRECT SEQUENCE
```

```
7903  032322                          13$:
7904  032322  005215                          INC     (R5)
7905
7906
7907                                  ;*************************************************************************
7908                                  ;TEST:215       125252 125252 SHIFTED BY @(4) = 177525 52525  PS = 10
7909                                  ;*************************************************************************
7910
7911  032324                          TST215:
7912  032324  012700  125252                  MOV     #125252,%0              ;LOAD R0 WITH 125252
7913  032330  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
7914  032334  000241                          CLC
7915  032336  073074  000000                  ASHC    @(4),%0         ;SHIFT R0,R0!1 BY @(4)
7916  032342  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7917  032346  122737  000010  025274         CMPB    #10,@#PSWORD    ;IS THE PS 10?
7918  032354  001401                          BEQ     11$
7919  032356  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
7920  032360  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7921  032364  001401                          BEQ     12$
7922  032366  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
7923  032370  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
7924  032374  001401                          BEQ     13$
7925  032376  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
7926  032400                          13$:
7927  032400  005215                          INC     (R5)
7928
7929
7930                                  ;*************************************************************************
7931                                  ;TEST:216       125252 125252 SHIFTED BY @(4)+ = 177525 52525  PS = 10
7932                                  ;*************************************************************************
7933
7934  032402                          TST216:
7935  032402  012700  125252                  MOV     #125252,%0              ;LOAD R0 WITH 125252
7936  032406  012701  125252                  MOV     #125252,%0!1    ;LOAD R0!1 WITH 125252
7937  032412  000241                          CLC
7938  032414  073034                          ASHC    @(4)+,%0                ;SHIFT R0,R0!1 BY @(4)+
7939  032416  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
7940  032422  122737  000010  025274         CMPB    #10,@#PSWORD    ;IS THE PS 10?
7941  032430  001401                          BEQ     11$
7942  032432  104000                          EMT                     ;THE PS IS NOT EQUAL TO 10
7943  032434  022700  177525          11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7944  032440  001401                          BEQ     12$
7945  032442  104000                          EMT                     ;R0 IS NOT EQUAL TO 177525
7946  032444  022701  052525          12$:    CMP     #52525,%0!1     ;IS THE RESULT 52525?
7947  032450  001401                          BEQ     13$
7948  032452  104000                          EMT                     ;R0!1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
7949  032454                          13$:
7950  032454  005215                          INC     (R5)
7951
7952
7953                                  ;*************************************************************************
7954                                  ;TEST:217       125252 125252 SHIFTED BY @-(4) = 177525 52525  PS = 10
7955                                  ;*************************************************************************
7956
7957  032456                          TST217:
7958  032456  012700  125252                  MOV     #125252,%0              ;LOAD R0 WITH 125252
```

```
7959  032462  012701  125252          MOV     #125252,%0:1    ;LOAD R0:1 WITH 125252
7960  032466  000241                  CLC
7961  032470  073054                  ASHC    @-(4),%0                ;SHIFT R0,R0:1 BY @-(4)
7962  032472  106737  025274          MFPS    @#PSWORD        ;SAVE PS
7963  032476  122737  000010  025274  CMPB    #10,@#PSWORD    ;IS THE PS 10?
7964  032504  001401                  BEQ     11$
7965  032506  104000                  EMT                     ;THE PS IS NOT EQUAL TO 10
7966  032510  022700  177525  11$:    CMP     #177525,%0      ;IS THE RESULT 177525?
7967  032514  001401                  BEQ     12$
7968  032516  104000                  EMT                     ;R0 IS NOT EQUAL TO 177525
7969  032520  022701  052525  12$:    CMP     #52525,%0:1     ;IS THE RESULT 52525?
7970  032524  001401                  BEQ     13$
7971  032526  104000                  EMT                     ;R0:1 IS NOT EQUAL TO 52525 OR INCORRECT SEQUENCE
7972  032530                  13$:
7973  032530  005215                  INC     (R5)
7974
7975
7976
7977
7978
7979
7980
7981
7982
```

```
                                        ;*********************************
                                        ;       MUL INSTRUCTION TESTS
                                        ;*********************************



                                        ;*************************************************************************
                                        ;TEST:220       MUL     1 * #0 = 0 0        PS = 4
                                        ;*************************************************************************

032532                                  TST220:
032532  012700  000001                          MOV     #1,%0           ;LOAD MULTIPLICAND WITH 1
032536  070027  000000                          MUL     #0,%0           ;MULTIPLY 1 * #0
032542  106737  025274                          MFPS    @#PSWORD        ;SAVE PS
032546  122737  000004  025274                  CMPB    #4,@#PSWORD     ;IS PS = 4
032554  001401                                  BEQ     11$
032556  104000                                  EMT                     ;PS IS WRONG
032560  022700  000000                  11$:    CMP     #0,%0           ;IS HIGH ORDER = 0
032564  001401                                  BEQ     12$
032566  104000                                  EMT                     ;HIGH ORDER IS WRONG
032570  022701  000000                  12$:    CMP     #0,%0!1         ;IS LOW ORDER = 0
032574  001401                                  BEQ     13$
032576  104000                                  EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
032600                                  13$:
032600  005215                                  INC     (R5)



                                        ;*************************************************************************
                                        ;TEST:221       MUL     -1 * #1 = -1 -1         PS = 10
                                        ;*************************************************************************

032602                                  TST221:
032602  012700  177777                          MOV     #-1,%0          ;LOAD MULTIPLICAND WITH -1
032606  070027  000001                          MUL     #1,%0           ;MULTIPLY -1 * #1
032612  106737  025274                          MFPS    @#PSWORD        ;SAVE PS
032616  122737  000010  025274                  CMPB    #10,@#PSWORD    ;IS PS = 10
032624  001401                                  BEQ     11$
032626  104000                                  EMT                     ;PS IS WRONG
032630  022700  177777                  11$:    CMP     #-1,%0          ;IS HIGH ORDER = -1
032634  001401                                  BEQ     12$
032636  104000                                  EMT                     ;HIGH ORDER IS WRONG
032640  022701  177777                  12$:    CMP     #-1,%0!1        ;IS LOW ORDER = -1
032644  001401                                  BEQ     13$
032646  104000                                  EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
032650                                  13$:
032650  005215                                  INC     (R5)



                                        ;*************************************************************************
                                        ;TEST:222       MUL     2 * #2 = 0 4        PS = 0
                                        ;*************************************************************************

032652                                  TST222:
032652  012702  000002                          MOV     #2,%2           ;LOAD MULTIPLICAND WITH 2
032656  070227  000002                          MUL     #2,%2           ;MULTIPLY 2 * #2
```

```
8039  032662  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8040  032666  122737  000000  025274          CMPB    #0,@#PSWORD     ;IS PS = 0
8041  032674  001401                          BEQ     11$
8042  032676  104000                          EMT                     ;PS IS WRONG
8043  032700  022702  000000          11$:    CMP     #0,%2           ;IS HIGH ORDER = 0
8044  032704  001401                          BEQ     12$
8045  032706  104000                          EMT                     ;HIGH ORDER IS WRONG
8046  032710  022703  000004          12$:    CMP     #4,%2!1         ;IS LOW ORDER = 4
8047  032714  001401                          BEQ     13$
8048  032716  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8049  032720                          13$:
8050  032720  005215                          INC     (R5)
8051
8052
8053                                  ;*************************************************************************
8054                                  ;TEST:223       MUL     1000 * #200 = 1 0       PS = 1
8055                                  ;*************************************************************************
8056
8057  032722                          TST223:
8058  032722  010501                          MOV     R5,R1           ;SAVE R5
8059  032724  012704  001000                  MOV     #1000,%4        ;LOAD MULTIPLICAND WITH 1000
8060  032730  070427  000200                  MUL     #200,%4         ;MULTIPLY 1000 * #200
8061  032734  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8062  032740  122737  000001  025274          CMPB    #1,@#PSWORD     ;IS PS = 1
8063  032746  001401                          BEQ     11$
8064  032750  104000                          EMT                     ;PS IS WRONG
8065  032752  022704  000001          11$:    CMP     #1,%4           ;IS HIGH ORDER = 1
8066  032756  001401                          BEQ     12$
8067  032760  104000                          EMT                     ;HIGH ORDER IS WRONG
8068  032762  022705  000000          12$:    CMP     #0,%4!1         ;IS LOW ORDER = 0
8069  032766  001401                          BEQ     13$
8070  032770  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8071  032772                          13$:
8072  032772  010105                          MOV     R1,R5           ;RESTORE R5
8073  032774  005215                          INC     (R5)
8074
8075
8076                                  ;*************************************************************************
8077                                  ;TEST:224       MUL     2 * #77777 = 0 177776       PS = 1
8078                                  ;*************************************************************************
8079
8080  032776                          TST224:
8081  032776  012700  000002                  MOV     #2,%0           ;LOAD MULTIPLICAND WITH 2
8082  033002  070027  077777                  MUL     #77777,%0       ;MULTIPLY 2 * #77777
8083  033006  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8084  033012  122737  000001  025274          CMPB    #1,@#PSWORD     ;IS PS = 1
8085  033020  001401                          BEQ     11$
8086  033022  104000                          EMT                     ;PS IS WRONG
8087  033024  022700  000000          11$:    CMP     #0,%0           ;IS HIGH ORDER = 0
8088  033030  001401                          BEQ     12$
8089  033032  104000                          EMT                     ;HIGH ORDER IS WRONG
8090  033034  022701  177776          12$:    CMP     #177776,%0!1    ;IS LOW ORDER = 177776
8091  033040  001401                          BEQ     13$
8092  033042  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8093  033044                          13$:
8094  033044  005215                          INC     (R5)
```

```
8095
8096
8097                                    ;*****************************************************************************
8098                                    ;TEST:225       MUL     7777 * #10 = 0 77770        PS = 0
8099                                    ;*****************************************************************************
8100
8101  033046                            TST225:
8102  033046  012702  007777                   MOV     #7777,%2        ;LOAD MULTIPLICAND WITH 7777
8103  033052  070227  000010                   MUL     #10,%2          ;MULTIPLY 7777 * #10
8104  033056  106737  025274                   MFPS    @#PSWORD        ;SAVE PS
8105  033062  122737  000000  025274           CMPB    #0,@#PSWORD     ;IS PS = 0
8106  033070  001401                           BEQ     11$
8107  033072  104000                           EMT                     ;PS IS WRONG
8108  033074  022702  000000            11$:   CMP     #0,%2           ;IS HIGH ORDER = 0
8109  033100  001401                           BEQ     12$
8110  033102  104000                           EMT                     ;HIGH ORDER IS WRONG
8111  033104  022703  077770            12$:   CMP     #77770,%2!1     ;IS LOW ORDER = 77770
8112  033110  001401                           BEQ     13$
8113  033112  104000                           EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8114  033114                            13$:
8115  033114  005215                           INC     (R5)
8116
8117
8118                                    ;*****************************************************************************
8119                                    ;TEST:226       MUL     77777 * #77777 = 37777 1        PS = 1
8120                                    ;*****************************************************************************
8121
8122  033116                            TST226:
8123  033116  010501                           MOV     R5,R1           ;SAVE R5
8124  033120  012704  077777                   MOV     #77777,%4       ;LOAD MULTIPLICAND WITH 77777
8125  033124  070427  077777                   MUL     #77777,%4       ;MULTIPLY 77777 * #77777
8126  033130  106737  025274                   MFPS    @#PSWORD        ;SAVE PS
8127  033134  122737  000001  025274           CMPB    #1,@#PSWORD     ;IS PS = 1
8128  033142  001401                           BEQ     11$
8129  033144  104000                           EMT                     ;PS IS WRONG
8130  033146  022704  037777            11$:   CMP     #37777,%4       ;IS HIGH ORDER = 37777
8131  033152  001401                           BEQ     12$
8132  033154  104000                           EMT                     ;HIGH ORDER IS WRONG
8133  033156  022705  000001            12$:   CMP     #1,%4!1         ;IS LOW ORDER = 1
8134  033162  001401                           BEQ     13$
8135  033164  104000                           EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8136  033166                            13$:
8137  033166  010105                           MOV     R1,R5           ;RESTORE R5
8138  033170  005215                           INC     (R5)
8139
8140
8141                                    ;*****************************************************************************
8142                                    ;TEST:227       MUL     -1 * #77777 = -1 100001         PS = 10
8143                                    ;*****************************************************************************
8144
8145  033172                            TST227:
8146  033172  012702  177777                   MOV     #-1,%2          ;LOAD MULTIPLICAND WITH -1
8147  033176  070227  077777                   MUL     #77777,%2       ;MULTIPLY -1 * #77777
8148  033202  106737  025274                   MFPS    @#PSWORD        ;SAVE PS
8149  033206  122737  000010  025274           CMPB    #10,@#PSWORD    ;IS PS = 10
8150  033214  001401                           BEQ     11$
```

```
8151  033216  104000                          EMT                     ;PS IS WRONG
8152  033220  022702  177777          11$:    CMP     #-1,%2          ;IS HIGH ORDER = -1
8153  033224  001401                          BEQ     12$
8154  033226  104000                          EMT                     ;HIGH ORDER IS WRONG
8155  033230  022703  100001          12$:    CMP     #100001,%2!1    ;IS LOW ORDER = 100001
8156  033234  001401                          BEQ     13$
8157  033236  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8158  033240                          13$:
8159  033240  005215                          INC     (R5)
8160
8161
8162                                  ;*********************************************************************
8163                                  ;TEST:230       MUL     -2 * #77777 = -1 2      PS = 11
8164                                  ;*********************************************************************
8165
8166  033242                          TST230:
8167  033242  012700  177776                  MOV     #-2,%0          ;LOAD MULTIPLICAND WITH -2
8168  033246  070027  077777                  MUL     #77777,%0       ;MULTIPLY -2 * #77777
8169  033252  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8170  033256  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8171  033264  001401                          BEQ     11$
8172  033266  104000                          EMT                     ;PS IS WRONG
8173  033270  022700  177777          11$:    CMP     #-1,%0          ;IS HIGH ORDER = -1
8174  033274  001401                          BEQ     12$
8175  033276  104000                          EMT                     ;HIGH ORDER IS WRONG
8176  033300  022701  000002          12$:    CMP     #2,%0!1         ;IS LOW ORDER = 2
8177  033304  001401                          BEQ     13$
8178  033306  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8179  033310                          13$:
8180  033310  005215                          INC     (R5)
8181
8182
8183                                  ;*********************************************************************
8184                                  ;TEST:231       MUL     125252 * #2 = -1 52524      PS = 11
8185                                  ;*********************************************************************
8186
8187  033312                          TST231:
8188  033312  012702  125252                  MOV     #125252,%2      ;LOAD MULTIPLICAND WITH 125252
8189  033316  070227  000002                  MUL     #2,%2           ;MULTIPLY 125252 * #2
8190  033322  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8191  033326  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8192  033334  001401                          BEQ     11$
8193  033336  104000                          EMT                     ;PS IS WRONG
8194  033340  022702  177777          11$:    CMP     #-1,%2          ;IS HIGH ORDER = -1
8195  033344  001401                          BEQ     12$
8196  033346  104000                          EMT                     ;HIGH ORDER IS WRONG
8197  033350  022703  052524          12$:    CMP     #52524,%2!1     ;IS LOW ORDER = 52524
8198  033354  001401                          BEQ     13$
8199  033356  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8200  033360                          13$:
8201  033360  005215                          INC     (R5)
8202
8203
8204                                  ;*********************************************************************
8205                                  ;TEST:232       MUL     125252 * #40000 = 165252 100000      PS = 11
8206                                  ;*********************************************************************
```

```
8207
8208  033362                                TST232:
8209  033362  010501                              MOV     R5,R1           ;SAVE R5
8210  033364  012704  125252                      MOV     #125252,%4      ;LOAD MULTIPLICAND WITH 125252
8211  033370  070427  040000                      MUL     #40000,%4       ;MULTIPLY 125252 * #40000
8212  033374  106737  025274                      MFPS    @#PSWORD        ;SAVE PS
8213  033400  122737  000011  025274              CMPB    #11,@#PSWORD    ;IS PS = 11
8214  033406  001401                              BEQ     11$
8215  033410  104000                              EMT                     ;PS IS WRONG
8216  033412  022704  165252          11$:        CMP     #165252,%4      ;IS HIGH ORDER = 165252
8217  033416  001401                              BEQ     12$
8218  033420  104000                              EMT                     ;HIGH ORDER IS WRONG
8219  033422  022705  100000          12$:        CMP     #100000,%4!1    ;IS LOW ORDER = 100000
8220  033426  001401                              BEQ     13$
8221  033430  104000                              EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8222  033432                                13$:
8223  033432  010105                              MOV     R1,R5           ;RESTORE R5
8224  033434  005215                              INC     (R5)
8225
8226
8227                                        ;*********************************************************************
8228                                        ;TEST:233       MUL     107070 * #107070 = 31222 26100      PS = 1
8229                                        ;*********************************************************************
8230
8231  033436                                TST233:
8232  033436  012700  107070                      MOV     #107070,%0      ;LOAD MULTIPLICAND WITH 107070
8233  033442  070027  107070                      MUL     #107070,%0      ;MULTIPLY 107070 * #107070
8234  033446  106737  025274                      MFPS    @#PSWORD        ;SAVE PS
8235  033452  122737  000001  025274              CMPB    #1,@#PSWORD     ;IS PS = 1
8236  033460  001401                              BEQ     11$
8237  033462  104000                              EMT                     ;PS IS WRONG
8238  033464  022700  031222          11$:        CMP     #31222,%0       ;IS HIGH ORDER = 31222
8239  033470  001401                              BEQ     12$
8240  033472  104000                              EMT                     ;HIGH ORDER IS WRONG
8241  033474  022701  026100          12$:        CMP     #26100,%0!1     ;IS LOW ORDER = 26100
8242  033500  001401                              BEQ     13$
8243  033502  104000                              EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8244  033504                                13$:
8245  033504  005215                              INC     (R5)
8246
8247
8248                                        ;*********************************************************************
8249                                        ;TEST:234       MUL     -1 * #1 = -1 -1·        PS = 10
8250                                        ;*********************************************************************
8251
8252  033506                                TST234:
8253  033506  012701  177777                      MOV     #-1,%1          ;LOAD MULTIPLICAND WITH -1
8254  033512  070127  000001                      MUL     #1,%1           ;MULTIPLY -1 * #1
8255  033516  106737  025274                      MFPS    @#PSWORD        ;SAVE PS
8256  033522  122737  000010  025274              CMPB    #10,@#PSWORD    ;IS PS = 10
8257  033530  001401                              BEQ     11$
8258  033532  104000                              EMT                     ;PS IS WRONG
8259  033534  022701  177777          11$:        CMP     #-1,%1          ;IS HIGH ORDER = -1
8260  033540  001401                              BEQ     12$
8261  033542  104000                              EMT                     ;HIGH ORDER IS WRONG
8262  033544  022701  177777          12$:        CMP     #-1,%1!1        ;IS LOW ORDER = -1
```

```
                        BEQ     13$
                        EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
                13$:
                        INC     (R5)


                ;*****************************************************************************
                ;TEST:235       MUL     -1 * #0 = 0 0           PS = 4
                ;*****************************************************************************

                TST235:
                        MOV     #-1,%3          ;LOAD MULTIPLICAND WITH -1
                        MUL     #0,%3           ;MULTIPLY -1 * #0
                        MFPS    @#PSWORD        ;SAVE PS
                        CMPB    #4,@#PSWORD     ;IS PS = 4
                        BEQ     11$
                        EMT                     ;PS IS WRONG
                11$:    CMP     #0,%3           ;IS HIGH ORDER = 0
                        BEQ     12$
                        EMT                     ;HIGH ORDER IS WRONG
                12$:    CMP     #0,%3!1         ;IS LOW ORDER = 0
                        BEQ     13$
                        EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
                13$:
                        INC     (R5)


                ;*****************************************************************************
                ;TEST:236       MUL     77777 * #100000 = 100000 100000         PS = 11
                ;*****************************************************************************

                TST236:
                        MOV     R5,R1           ;SAVE R5
                        MOV     #77777,%5       ;LOAD MULTIPLICAND WITH 77777
                        MUL     #100000,%5      ;MULTIPLY 77777 * #100000
                        MFPS    @#PSWORD        ;SAVE PS
                        CMPB    #11,@#PSWORD    ;IS PS = 11
                        BEQ     11$
                        EMT                     ;PS IS WRONG
                11$:    CMP     #100000,%5      ;IS HIGH ORDER = 100000
                        BEQ     12$
                        EMT                     ;HIGH ORDER IS WRONG
                12$:    CMP     #100000,%5!1    ;IS LOW ORDER = 100000
                        BEQ     13$
                        EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
                13$:
                        MOV     R1,R5           ;RESTORE R5
                        INC     (R5)


                ;*****************************************************************************
                ;TEST:237       MUL     -1 * #77777 = 100001 100001         PS = 10
                ;*****************************************************************************

                TST237:
                        MOV     #-1,%1          ;LOAD MULTIPLICAND WITH -1
```

```
 8319  033706  070127  077777                 MUL     #77777,%1       ;MULTIPLY -1 * #77777
 8320  033712  106737  025274                 MFPS    @#PSWORD        ;SAVE PS
 8321  033716  122737  000010  025274         CMPB    #10,@#PSWORD    ;IS PS = 10
 8322  033724  001401                         BEQ     11$
 8323  033726  104000                         EMT                     ;PS IS WRONG
 8324  033730  022701  100001         11$:    CMP     #100001,%1      ;IS HIGH ORDER = 100001
 8325  033734  001401                         BEQ     12$
 8326  033736  104000                         EMT                     ;HIGH ORDER IS WRONG
 8327  033740  022701  100001         12$:    CMP     #100001,%1!1    ;IS LOW ORDER = 100001
 8328  033744  001401                         BEQ     13$
 8329  033746  104000                         EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
 8330  033750                         13$:
 8331  033750  005215                         INC     (R5)
 8332
 8333
 8334                                 ;*****************************************************************************
 8335                                 ;TEST:240       MUL       77777 * #77777 = 1 1        PS = 1
 8336                                 ;*****************************************************************************
 8337
 8338  033752                         TST240:
 8339  033752  012703  077777                 MOV     #77777,%3       ;LOAD MULTIPLICAND WITH 77777
 8340  033756  070327  077777                 MUL     #77777,%3       ;MULTIPLY 77777 * #77777
 8341  033762  106737  025274                 MFPS    @#PSWORD        ;SAVE PS
 8342  033766  122737  000001  025274         CMPB    #1,@#PSWORD     ;IS PS = 1
 8343  033774  001401                         BEQ     11$
 8344  033776  104000                         EMT                     ;PS IS WRONG
 8345  034000  022703  000001         11$:    CMP     #1,%3           ;IS HIGH ORDER = 1
 8346  034004  001401                         BEQ     12$
 8347  034006  104000                         EMT                     ;HIGH ORDER IS WRONG
 8348  034010  022703  000001         12$:    CMP     #1,%3!1         ;IS LOW ORDER = 1
 8349  034014  001401                         BEQ     13$
 8350  034016  104000                         EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
 8351  034020                         13$:
 8352  034020  005215                         INC     (R5)
 8353
 8354
 8355                                 ;*****************************************************************************
 8356                                 ;TEST:241       MUL       2 * #2 = 4 4        PS = 0
 8357                                 ;*****************************************************************************
 8358
 8359  034022                         TST241:
 8360  034022  010501                         MOV     R5,R1           ;SAVE R5
 8361  034024  012705  000002                 MOV     #2,%5           ;LOAD MULTIPLICAND WITH 2
 8362  034030  070527  000002                 MUL     #2,%5           ;MULTIPLY 2 * #2
 8363  034034  106737  025274                 MFPS    @#PSWORD        ;SAVE PS
 8364  034040  122737  000000  025274         CMPB    #0,@#PSWORD     ;IS PS = 0
 8365  034046  001401                         BEQ     11$
 8366  034050  104000                         EMT                     ;PS IS WRONG
 8367  034052  022705  000004         11$:    CMP     #4,%5           ;IS HIGH ORDER = 4
 8368  034056  001401                         BEQ     12$
 8369  034060  104000                         EMT                     ;HIGH ORDER IS WRONG
 8370  034062  022705  000004         12$:    CMP     #4,%5!1         ;IS LOW ORDER = 4
 8371  034066  001401                         BEQ     13$
 8372  034070  104000                         EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
 8373  034072                         13$:
 8374  034072  010105                         MOV     R1,R5           ;RESTORE R5
```

```
8375  034074  005215                          INC     (R5)
8376
8377
8378  034076  012702  040000                  MOV     #40000,%2
8379  034102  012703  025322                  MOV     #S5,%3
8380  034106  012704  025324                  MOV     #S6,%4
8381
8382                                  ;*****************************************************************************
8383                                  ;TEST:242       MUL     125252 * S5 = 165252 100000     PS = 11
8384                                  ;*****************************************************************************
8385
8386  034112                          TST242:
8387  034112  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8388  034116  070067  171200                  MUL     S5,%0           ;MULTIPLY 125252 * S5
8389  034122  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8390  034126  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8391  034134  001401                          BEQ     11$
8392  034136  104000                          EMT                     ;PS IS WRONG
8393  034140  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8394  034144  001401                          BEQ     12$
8395  034146  104000                          EMT                     ;HIGH ORDER IS WRONG
8396  034150  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8397  034154  001401                          BEQ     13$
8398  034156  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8399  034160                          13$:
8400  034160  005215                          INC     (R5)
8401
8402
8403                                  ;*****************************************************************************
8404                                  ;TEST:243       MUL     125252 * @S6 = 165252 100000     PS = 11
8405                                  ;*****************************************************************************
8406
8407  034162                          TST243:
8408  034162  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8409  034166  070077  171132                  MUL     @S6,%0          ;MULTIPLY 125252 * @S6
8410  034172  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8411  034176  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8412  034204  001401                          BEQ     11$
8413  034206  104000                          EMT                     ;PS IS WRONG
8414  034210  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8415  034214  001401                          BEQ     12$
8416  034216  104000                          EMT                     ;HIGH ORDER IS WRONG
8417  034220  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8418  034224  001401                          BEQ     13$
8419  034226  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8420  034230                          13$:
8421  034230  005215                          INC     (R5)
8422
8423
8424                                  ;*****************************************************************************
8425                                  ;TEST:244       MUL     125252 * @#S5 = 165252 100000     PS = 11
8426                                  ;*****************************************************************************
8427
8428  034232                          TST244:
8429  034232  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8430  034236  070037  025322                  MUL     @#S5,%0         ;MULTIPLY 125252 * @#S5
```

```
8431  034242  106737  025274            MFPS    @#PSWORD        ;SAVE PS
8432  034246  122737  000011  025274    CMPB    #11,@#PSWORD    ;IS PS = 11
8433  034254  001401                    BEQ     11$
8434  034256  104000                    EMT                     ;PS IS WRONG
8435  034260  022700  165252    11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8436  034264  001401                    BEQ     12$
8437  034266  104000                    EMT                     ;HIGH ORDER IS WRONG
8438  034270  022701  100000    12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8439  034274  001401                    BEQ     13$
8440  034276  104000                    EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8441  034300                    13$:
8442  034300  005215                    INC     (R5)
8443
8444
8445                            ;*****************************************************************************
8446                            ;TEST:245       MUL     125252 * %2 = 165252 100000     PS = 11
8447                            ;*****************************************************************************
8448
8449  034302                    TST245:
8450  034302  012700  125252            MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8451  034306  070002                    MUL     %2,%0           ;MULTIPLY 125252 * %2
8452  034310  106737  025274            MFPS    @#PSWORD        ;SAVE PS
8453  034314  122737  000011  025274    CMPB    #11,@#PSWORD    ;IS PS = 11
8454  034322  001401                    BEQ     11$
8455  034324  104000                    EMT                     ;PS IS WRONG
8456  034326  022700  165252    11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8457  034332  001401                    BEQ     12$
8458  034334  104000                    EMT                     ;HIGH ORDER IS WRONG
8459  034336  022701  100000    12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8460  034342  001401                    BEQ     13$
8461  034344  104000                    EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8462  034346                    13$:
8463  034346  005215                    INC     (R5)
8464
8465
8466                            ;*****************************************************************************
8467                            ;TEST:246       MUL     125252 * (3)+ = 165252 100000     PS = 11
8468                            ;*****************************************************************************
8469
8470  034350                    TST246:
8471  034350  012700  125252            MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8472  034354  070023                    MUL     (3)+,%0         ;MULTIPLY 125252 * (3)+
8473  034356  106737  025274            MFPS    @#PSWORD        ;SAVE PS
8474  034362  122737  000011  025274    CMPB    #11,@#PSWORD    ;IS PS = 11
8475  034370  001401                    BEQ     11$
8476  034372  104000                    EMT                     ;PS IS WRONG
8477  034374  022700  165252    11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8478  034400  001401                    BEQ     12$
8479  034402  104000                    EMT                     ;HIGH ORDER IS WRONG
8480  034404  022701  100000    12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8481  034410  001401                    BEQ     13$
8482  034412  104000                    EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8483  034414                    13$:
8484  034414  005215                    INC     (R5)
8485
8486
```

```
8487
8488                    ;*****************************************************************************
8489                    ;TEST:247       MUL     125252 * -(3) = 165252 100000       PS = 11
8490                    ;*****************************************************************************
8491   034416
8492   034416  012700  125252          TST247: MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8493   034422  070043                          MUL     -(3),%0         ;MULTIPLY 125252 * -(3)
8494   034424  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8495   034430  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8496   034436  001401                          BEQ     11$
8497   034440  104000                          EMT                     ;PS IS WRONG
8498   034442  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8499   034446  001401                          BEQ     12$
8500   034450  104000                          EMT                     ;HIGH ORDER IS WRONG
8501   034452  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8502   034456  001401                          BEQ     13$
8503   034460  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8504   034462                          13$:
8505   034462  005215                          INC     (R5)
8506
8507                    ;*****************************************************************************
8508                    ;TEST:250       MUL     125252 * 2(4) = 165252 100000       PS = 11
8509                    ;*****************************************************************************
8510
8511   034464          TST250: MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8512   034464  012700  125252          MUL     2(4),%0         ;MULTIPLY 125252 * 2(4)
8513   034470  070064  000002          MFPS    @#PSWORD        ;SAVE PS
8514   034474  106737  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8515   034500  122737  000011  025274  BEQ     11$
8516   034506  001401                  EMT                     ;PS IS WRONG
8517   034510  104000          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8518   034512  022700  165252          BEQ     12$
8519   034516  001401                  EMT                     ;HIGH ORDER IS WRONG
8520   034520  104000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8521   034522  022701  100000          BEQ     13$
8522   034526  001401                  EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8523   034530  104000          13$:
8524   034532                  13$:    INC     (R5)
8525   034532  005215
8526
8527                    ;*****************************************************************************
8528                    ;TEST:251       MUL     125252 * @(4) = 165252 100000       PS = 11
8529                    ;*****************************************************************************
8530
8531   034534          TST251: MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8532   034534  012700  125252          MUL     @(4),%0         ;MULTIPLY 125252 * @(4)
8533   034540  070074  000000          MFPS    @#PSWORD        ;SAVE PS
8534   034544  106737  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8535   034550  122737  000011  025274  BEQ     11$
8536   034556  001401                  EMT                     ;PS IS WRONG
8537   034560  104000          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8538   034562  022700  165252          BEQ     12$
8539   034566  001401                  EMT                     ;HIGH ORDER IS WRONG
8540   034570  104000
8541
8542
```

```
8543  034572  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8544  034576  001401                          BEQ     13$
8545  034600  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8546  034602                          13$:
8547  034602  005215                          INC     (R5)
8548
8549
8550                                  ;*************************************************************************
8551                                  ;TEST:252       MUL     125252 * @(4)+ = 165252 100000          PS = 11
8552                                  ;*************************************************************************
8553
8554  034604                          TST252:
8555  034604  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8556  034610  070034                          MUL     @(4)+,%0        ;MULTIPLY 125252 * @(4)+
8557  034612  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8558  034616  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8559  034624  001401                          BEQ     11$
8560  034626  104000                          EMT                     ;PS IS WRONG
8561  034630  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8562  034634  001401                          BEQ     12$
8563  034636  104000                          EMT                     ;HIGH ORDER IS WRONG
8564  034640  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8565  034644  001401                          BEQ     13$
8566  034646  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8567  034650                          13$:
8568  034650  005215                          INC     (R5)
8569
8570
8571                                  ;*************************************************************************
8572                                  ;TEST:253       MUL     125252 * @-(4) = 165252 100000          PS = 11
8573                                  ;*************************************************************************
8574
8575  034652                          TST253:
8576  034652  012700  125252                  MOV     #125252,%0      ;LOAD MULTIPLICAND WITH 125252
8577  034656  070054                          MUL     @-(4),%0        ;MULTIPLY 125252 * @-(4)
8578  034660  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8579  034664  122737  000011  025274          CMPB    #11,@#PSWORD    ;IS PS = 11
8580  034672  001401                          BEQ     11$
8581  034674  104000                          EMT                     ;PS IS WRONG
8582  034676  022700  165252          11$:    CMP     #165252,%0      ;IS HIGH ORDER = 165252
8583  034702  001401                          BEQ     12$
8584  034704  104000                          EMT                     ;HIGH ORDER IS WRONG
8585  034706  022701  100000          12$:    CMP     #100000,%0!1    ;IS LOW ORDER = 100000
8586  034712  001401                          BEQ     13$
8587  034714  104000                          EMT                     ;LOW ORDER IS WRONG OR WRONG SEQUENCE
8588  034716                          13$:
8589  034716  005215                          INC     (R5)
8590
8591
```

```
8592                                    ;*********************************
8593                                    ;     DIV INSTRUCTION TESTS
8594                                    ;*********************************
8595
8596
8597
8598                                    ;*************************************************************************
8599                                    ;TEST:254      DIV     0 4 / #2 = 2       REM = 0      PS = 0
8600                                    ;*************************************************************************
8601
8602  034720                            TST254:
8603  034720  012700  000000                   MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
8604  034724  012701  000004                   MOV     #4,%0+1         ;LOAD LOW ORDER WITH 4
8605  034730  071027  000002                   DIV     #2,%0           ;DIVIDE BY #2
8606  034734  106737  025274                   MFPS    @#PSWORD        ;SAVE PS
8607  034740  122737  000000  025274           CMPB    #0,@#PSWORD     ;IS PS = 0
8608  034746  001401                           BEQ     11$
8609  034750  104000                           EMT                     ;PS IS WRONG
8610  034752  022700  000002           11$:    CMP     #2,%0           ;IS QUOTIENT = 2
8611  034756  001401                           BEQ     12$
8612  034760  104000                           EMT                     ;QUOTIENT IS WRONG
8613  034762  022701  000000           12$:    CMP     #0,%0+1         ;IS REMAINDER = 0
8614  034766  001401                           BEQ     13$
8615  034770  104000                           EMT                     ;WRONG REMAINDER
8616  034772                            13$:
8617  034772  005215                           INC     (R5)
8618
8619                                    ;*************************************************************************
8620                                    ;TEST:255      DIV     -1 -9. / #3 = -3       REM = 0      PS = 10
8621                                    ;*************************************************************************
8622
8623  034774                            TST255:
8624  034774  012702  177777                   MOV     #-1,%2          ;LOAD HIGH ORDER WITH -1
8625  035000  012703  177767                   MOV     #-9.,%2+1       ;LOAD LOW ORDER WITH -9.
8626  035004  071227  000003                   DIV     #3,%2           ;DIVIDE BY #3
8627  035010  106737  025274                   MFPS    @#PSWORD        ;SAVE PS
8628  035014  122737  000010  025274           CMPB    #10,@#PSWORD    ;IS PS = 10
8629  035022  001401                           BEQ     11$
8630  035024  104000                           EMT                     ;PS IS WRONG
8631  035026  022702  177775           11$:    CMP     #-3,%2          ;IS QUOTIENT = -3
8632  035032  001401                           BEQ     12$
8633  035034  104000                           EMT                     ;QUOTIENT IS WRONG
8634  035036  022703  000000           12$:    CMP     #0,%2+1         ;IS REMAINDER = 0
8635  035042  001401                           BEQ     13$
8636  035044  104000                           EMT                     ;WRONG REMAINDER
8637  035046                            13$:
8638  035046  005215                           INC     (R5)
8639
8640                                    ;*************************************************************************
8641                                    ;TEST:256      DIV     0 9. / #2 = 4       REM = 1      PS = 0
8642                                    ;*************************************************************************
8643
8644  035050                            TST256:
8645  035050  010501                           MOV     R5,R1           ;SAVE R5
8646  035052  012704  000000                   MOV     #0,%4           ;LOAD HIGH ORDER WITH 0
8647  035056  012705  000011                   MOV     #9.,%4+1        ;LOAD LOW ORDER WITH 9.
```

```
8648  035062  071427  000002                  DIV     #2,%4           ;DIVIDE BY #2
8649  035066  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8650  035072  122737  000000  025274          CMPB    #0,@#PSWORD     ;IS PS = 0
8651  035100  001401                          BEQ     11$
8652  035102  104000                          EMT                     ;PS IS WRONG
8653  035104  022704  000004          11$:    CMP     #4,%4           ;IS QUOTIENT = 4
8654  035110  001401                          BEQ     12$
8655  035112  104000                          EMT                     ;QUOTIENT IS WRONG
8656  035114  022705  000001          12$:    CMP     #1,%4+1         ;IS REMAINDER = 1
8657  035120  001401                          BEQ     13$
8658  035122  104000                          EMT                     ;WRONG REMAINDER
8659  035124                          13$:
8660  035124  010105                          MOV     R1,R5           ;RESTORE R5
8661  035126  005215                          INC     (R5)
8662
8663                                  ;*******************************************************************************
8664                                  ;TEST:257       DIV     -1 -9. / #2 = -4        REM = -1        PS = 10
8665                                  ;*******************************************************************************
8666
8667  035130                          TST257:
8668  035130  012700  177777                  MOV     #-1,%0          ;LOAD HIGH ORDER WITH -1
8669  035134  012701  177767                  MOV     #-9.,%0+1       ;LOAD LOW ORDER WITH -9.
8670  035140  071027  000002                  DIV     #2,%0           ;DIVIDE BY #2
8671  035144  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8672  035150  122737  000010  025274          CMPB    #10,@#PSWORD    ;IS PS = 10
8673  035156  001401                          BEQ     11$
8674  035160  104000                          EMT                     ;PS IS WRONG
8675  035162  022700  177774          11$:    CMP     #-4,%0          ;IS QUOTIENT = -4
8676  035166  001401                          BEQ     12$
8677  035170  104000                          EMT                     ;QUOTIENT IS WRONG
8678  035172  022701  177777          12$:    CMP     #-1,%0+1        ;IS REMAINDER = -1
8679  035176  001401                          BEQ     13$
8680  035200  104000                          EMT                     ;WRONG REMAINDER
8681  035202                          13$:
8682  035202  005215                          INC     (R5)
8683
8684                                  ;*******************************************************************************
8685                                  ;TEST:260       DIV     0 2 / #-3 = 0           REM = 2         PS = 4
8686                                  ;*******************************************************************************
8687
8688  035204                          TST260:
8689  035204  012702  000000                  MOV     #0,%2           ;LOAD HIGH ORDER WITH 0
8690  035210  012703  000002                  MOV     #2,%2+1         ;LOAD LOW ORDER WITH 2
8691  035214  071227  177775                  DIV     #-3,%2          ;DIVIDE BY #-3
8692  035220  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8693  035224  122737  000004  025274          CMPB    #4,@#PSWORD     ;IS PS = 4
8694  035232  001401                          BEQ     11$
8695  035234  104000                          EMT                     ;PS IS WRONG
8696  035236  022702  000000          11$:    CMP     #0,%2           ;IS QUOTIENT = 0
8697  035242  001401                          BEQ     12$
8698  035244  104000                          EMT                     ;QUOTIENT IS WRONG
8699  035246  022703  000002          12$:    CMP     #2,%2+1         ;IS REMAINDER = 2
8700  035252  001401                          BEQ     13$
8701  035254  104000                          EMT                     ;WRONG REMAINDER
8702  035256                          13$:
8703  035256  005215                          INC     (R5)
```

```
                        ;*********************************************************************
                        ;TEST:261      DIV     -1 -2 / #3 = 0       REM = -2      PS = 4
                        ;*********************************************************************

035260                  TST261:
035260  010501                  MOV     R5,R1           ;SAVE R5
035262  012704  177777          MOV     #-1,%4          ;LOAD HIGH ORDER WITH -1
035266  012705  177776          MOV     #-2,%4+1        ;LOAD LOW ORDER WITH -2
035272  071427  000003          DIV     #3,%4           ;DIVIDE BY #3
035276  106737  025274          MFPS    @#PSWORD        ;SAVE PS
035302  122737  000004  025274  CMPB    #4,@#PSWORD     ;IS PS = 4
035310  001401                  BEQ     11$
035312  104000                  EMT                     ;PS IS WRONG
035314  022704  000000  11$:    CMP     #0,%4           ;IS QUOTIENT = 0
035320  001401                  BEQ     12$
035322  104000                  EMT                     ;QUOTIENT IS WRONG
035324  022705  177776  12$:    CMP     #-2,%4+1        ;IS REMAINDER = -2
035330  001401                  BEQ     13$
035332  104000                  EMT                     ;WRONG REMAINDER
035334                  13$:
035334  010105                  MOV     R1,R5           ;RESTORE R5
035336  005215                  INC     (R5)

                        ;*********************************************************************
                        ;TEST:262      DIV     -1 -1 / #1 = -1       REM = 0       PS = 10
                        ;*********************************************************************

035340                  TST262:
035340  012700  177777          MOV     #-1,%0          ;LOAD HIGH ORDER WITH -1
035344  012701  177777          MOV     #-1,%0+1        ;LOAD LOW ORDER WITH -1
035350  071027  000001          DIV     #1,%0           ;DIVIDE BY #1
035354  106737  025274          MFPS    @#PSWORD        ;SAVE PS
035360  122737  000010  025274  CMPB    #10,@#PSWORD    ;IS PS = 10
035366  001401                  BEQ     11$
035370  104000                  EMT                     ;PS IS WRONG
035372  022700  177777  11$:    CMP     #-1,%0          ;IS QUOTIENT = -1
035376  001401                  BEQ     12$
035400  104000                  EMT                     ;QUOTIENT IS WRONG
035402  022701  000000  12$:    CMP     #0,%0+1         ;IS REMAINDER = 0
035406  001401                  BEQ     13$
035410  104000                  EMT                     ;WRONG REMAINDER
035412                  13$:
035412  005215                  INC     (R5)

                        ;*********************************************************************
                        ;TEST:263      DIV     0 0 / #1 = 0       REM = 0       PS = 4
                        ;*********************************************************************

035414                  TST263:
035414  012700  000000          MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
035420  012701  000000          MOV     #0,%0+1         ;LOAD LOW ORDER WITH 0
035424  071027  000001          DIV     #1,%0           ;DIVIDE BY #1
035430  106737  025274          MFPS    @#PSWORD        ;SAVE PS
035434  122737  000004  025274  CMPB    #4,@#PSWORD     ;IS PS = 4
035442  001401                  BEQ     11$
```

```
8760  035444  104000                        EMT                     ;PS IS WRONG
8761  035446  022700  000000          11$:  CMP     #0,%0             ;IS QUOTIENT = 0
8762  035452  001401                        BEQ     12$
8763  035454  104000                        EMT                     ;QUOTIENT IS WRONG
8764  035456  022701  000000          12$:  CMP     #0,%0+1           ;IS REMAINDER = 0
8765  035462  001401                        BEQ     13$
8766  035464  104000                        EMT                     ;WRONG REMAINDER
8767  035466                          13$:
8768  035466  005215                        INC     (R5)
8769
8770                                  ;*****************************************************************************
8771                                  ;TEST:264       DIV     -1 125252 / #2 = 152525       REM = 0       PS = 10
8772                                  ;*****************************************************************************
8773
8774  035470                          TST264:
8775  035470  012702  177777                MOV     #-1,%2            ;LOAD HIGH ORDER WITH -1
8776  035474  012703  125252                MOV     #125252,%2+1      ;LOAD LOW ORDER WITH 125252
8777  035500  071227  000002                DIV     #2,%2             ;DIVIDE BY #2
8778  035504  106737  025274                MFPS    @#PSWORD          ;SAVE PS
8779  035510  122737  000010  025274        CMPB    #10,@#PSWORD      ;IS PS = 10
8780  035516  001401                        BEQ     11$
8781  035520  104000                        EMT                     ;PS IS WRONG
8782  035522  022702  152525          11$:  CMP     #152525,%2        ;IS QUOTIENT = 152525
8783  035526  001401                        BEQ     12$
8784  035530  104000                        EMT                     ;QUOTIENT IS WRONG
8785  035532  022703  000000          12$:  CMP     #0,%2+1           ;IS REMAINDER = 0
8786  035536  001401                        BEQ     13$
8787  035540  104000                        EMT                     ;WRONG REMAINDER
8788  035542                          13$:
8789  035542  005215                        INC     (R5)
8790
8791                                  ;*****************************************************************************
8792                                  ;TEST:265       DIV     -1 -1 / #-1 = 1       REM = 0       PS = 0
8793                                  ;*****************************************************************************
8794
8795  035544                          TST265:
8796  035544  010501                        MOV     R5,R1             ;SAVE R5
8797  035546  012704  177777                MOV     #-1,%4            ;LOAD HIGH ORDER WITH -1
8798  035552  012705  177777                MOV     #-1,%4+1          ;LOAD LOW ORDER WITH -1
8799  035556  071427  177777                DIV     #-1,%4            ;DIVIDE BY #-1
8800  035562  106737  025274                MFPS    @#PSWORD          ;SAVE PS
8801  035566  122737  000000  025274        CMPB    #0,@#PSWORD       ;IS PS = 0
8802  035574  001401                        BEQ     11$
8803  035576  104000                        EMT                     ;PS IS WRONG
8804  035600  022704  000001          11$:  CMP     #1,%4             ;IS QUOTIENT = 1
8805  035604  001401                        BEQ     12$
8806  035606  104000                        EMT                     ;QUOTIENT IS WRONG
8807  035610  022705  000000          12$:  CMP     #0,%4+1           ;IS REMAINDER = 0
8808  035614  001401                        BEQ     13$
8809  035616  104000                        EMT                     ;WRONG REMAINDER
8810  035620                          13$:
8811  035620  010105                        MOV     R1,R5             ;RESTORE R5
8812  035622  005215                        INC     (R5)
8813
8814                                  ;*****************************************************************************
8815                                  ;TEST:266       DIV     25253 1 / #125252 = 100000       REM = 1       PS = 10
```

```
8816
8817
8818  035624
8819  035624  012700  025253                TST266:
8820  035630  012701  000001                        MOV     #25253,%0       ;LOAD HIGH ORDER WITH 25253
8821  035634  071027  125252                        MOV     #1,%0+1         ;LOAD LOW ORDER WITH 1
8822  035640  106737  025274                        DIV     #125252,%0      ;DIVIDE BY #125252
8823  035644  122737  000010  025274                MFPS    @#PSWORD        ;SAVE PS
8824  035652  001401                                CMPB    #10,@#PSWORD    ;IS PS = 10
8825  035654  104000                                BEQ     11$
8826  035656  022700  100000                        EMT                     ;PS IS WRONG
8827  035662  001401                        11$:    CMP     #100000,%0      ;IS QUOTIENT = 100000
8828  035664  104000                                BEQ     12$
8829  035666  022701  000001                        EMT                     ;QUOTIENT IS WRONG
8830  035672  001401                        12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
8831  035674  104000                                BEQ     13$
8832  035676                                        EMT                     ;WRONG REMAINDER
8833  035676  005215                        13$:
8834                                                INC     (R5)
8835                                        ;*****************************************************************************
8836                                        ;TEST:267       DIV     37777 77777 / #77777 = 77777       REM = 77776      PS = 0
8837                                        ;*****************************************************************************
8838
8839  035700                                TST267:
8840  035700  012702  037777                        MOV     #37777,%2       ;LOAD HIGH ORDER WITH 37777
8841  035704  012703  077777                        MOV     #77777,%2+1     ;LOAD LOW ORDER WITH 77777
8842  035710  071227  077777                        DIV     #77777,%2       ;DIVIDE BY #77777
8843  035714  106737  025274                        MFPS    @#PSWORD        ;SAVE PS
8844  035720  122737  000000  025274                CMPB    #0,@#PSWORD     ;IS PS = 0
8845  035726  001401                                BEQ     11$
8846  035730  104000                                EMT                     ;PS IS WRONG
8847  035732  022702  077777                11$:    CMP     #77777,%2       ;IS QUOTIENT = 77777
8848  035736  001401                                BEQ     12$
8849  035740  104000                                EMT                     ;QUOTIENT IS WRONG
8850  035742  022703  077776                12$:    CMP     #77776,%2+1     ;IS REMAINDER = 77776
8851  035746  001401                                BEQ     13$
8852  035750  104000                                EMT                     ;WRONG REMAINDER
8853  035752                                13$:
8854  035752  005215                                INC     (R5)
8855
8856                                        ;*****************************************************************************
8857                                        ;TEST:270       DIV     0 100000 / #2 = 40000       REM = 0      PS = 0
8858                                        ;*****************************************************************************
8859
8860  035754                                TST270:
8861  035754  010501                                MOV     R5,R1           ;SAVE R5
8862  035756  012704  000000                        MOV     #0,%4           ;LOAD HIGH ORDER WITH 0
8863  035762  012705  100000                        MOV     #100000,%4+1    ;LOAD LOW ORDER WITH 100000
8864  035766  071427  000002                        DIV     #2,%4           ;DIVIDE BY #2
8865  035772  106737  025274                        MFPS    @#PSWORD        ;SAVE PS
8866  035776  122737  000000  025274                CMPB    #0,@#PSWORD     ;IS PS = 0
8867  036004  001401                                BEQ     11$
8868  036006  104000                                EMT                     ;PS IS WRONG
8869  036010  022704  040000                11$:    CMP     #40000,%4       ;IS QUOTIENT = 40000
8870  036014  001401                                BEQ     12$
8871  036016  104000                                EMT                     ;QUOTIENT IS WRONG
```

```
8872  036020  022705  000000          12$:    CMP     #0,%4+1         ;IS REMAINDER = 0
8873  036024  001401                          BEQ     13$
8874  036026  104000                          EMT                     ;WRONG REMAINDER
8875  036030                        13$:
8876  036030  010105                          MOV     R1,R5           ;RESTORE R5
8877  036032  005215                          INC     (R5)
8878
8879                                  ;*****************************************************************************
8880                                  ;TEST:271       DIV     177777 77777 / #177776 = 40000       REM = 177777      PS = 0
8881                                  ;*****************************************************************************
8882
8883  036034                        TST271:
8884  036034  012700  177777                 MOV     #177777,%0      ;LOAD HIGH ORDER WITH 177777
8885  036040  012701  077777                 MOV     #77777,%0+1     ;LOAD LOW ORDER WITH 77777
8886  036044  071027  177776                 DIV     #177776,%0      ;DIVIDE BY #177776
8887  036050  106737  025274                 MFPS    @#PSWORD        ;SAVE PS
8888  036054  122737  000000  025274        CMPB    #0,@#PSWORD     ;IS PS = 0
8889  036062  001401                          BEQ     11$
8890  036064  104000                          EMT                     ;PS IS WRONG
8891  036066  022700  040000          11$:    CMP     #40000,%0       ;IS QUOTIENT = 40000
8892  036072  001401                          BEQ     12$
8893  036074  104000                          EMT                     ;QUOTIENT IS WRONG
8894  036076  022701  177777          12$:    CMP     #177777,%0+1    ;IS REMAINDER = 177777
8895  036102  001401                          BEQ     13$
8896  036104  104000                          EMT                     ;WRONG REMAINDER
8897  036106                        13$:
8898  036106  005215                          INC     (R5)
8899
8900                                  ;*****************************************************************************
8901                                  ;TEST:272       DIV     0 52525 / #52525 = 1        REM = 0       PS = 0
8902                                  ;*****************************************************************************
8903
8904  036110                        TST272:
8905  036110  012702  000000                 MOV     #0,%2           ;LOAD HIGH ORDER WITH 0
8906  036114  012703  052525                 MOV     #52525,%2+1     ;LOAD LOW ORDER WITH 52525
8907  036120  071227  052525                 DIV     #52525,%2       ;DIVIDE BY #52525
8908  036124  106737  025274                 MFPS    @#PSWORD        ;SAVE PS
8909  036130  122737  000000  025274        CMPB    #0,@#PSWORD     ;IS PS = 0
8910  036136  001401                          BEQ     11$
8911  036140  104000                          EMT                     ;PS IS WRONG
8912  036142  022702  000001          11$:    CMP     #1,%2           ;IS QUOTIENT = 1
8913  036146  001401                          BEQ     12$
8914  036150  104000                          EMT                     ;QUOTIENT IS WRONG
8915  036152  022703  000000          12$:    CMP     #0,%2+1         ;IS REMAINDER = 0
8916  036156  001401                          BEQ     13$
8917  036160  104000                          EMT                     ;WRONG REMAINDER
8918  036162                        13$:
8919  036162  005215                          INC     (R5)
8920
8921                                  ;*****************************************************************************
8922                                  ;TEST:273       DIV     0 77777 / #0 = DUMMY        REM = DUMMY       PS = 3
8923                                  ;*****************************************************************************
8924
8925  036164                        TST273:
8926  036164  010501                          MOV     R5,R1           ;SAVE R5
8927  036166  012704  000000                 MOV     #0,%4           ;LOAD HIGH ORDER WITH 0
```

```
8928  036172  012705  077777                  MOV     #77777,%4+1     ;LOAD LOW ORDER WITH 77777
8929  036176  071427  000000                  DIV     #0,%4           ;DIVIDE BY #0
8930  036202  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8931  036206  042737  000014  025274          BIC     #14,@#PSWORD
8932  036214  122737  000003  025274          CMPB    #3,@#PSWORD     ;IS PS = 3
8933  036222  001401                          BEQ     13$
8934  036224  104000                          EMT                     ;PS IS WRONG
8935  036226                          13$:
8936  036226  010105                          MOV     R1,R5           ;RESTORE R5
8937  036230  005215                          INC     (R5)
8938
8939                                  ;*****************************************************************************
8940                                  ;TEST:274       DIV     77777 177777 / #2 = DUMMY       REM = DUMMY     PS = 2
8941                                  ;*****************************************************************************
8942
8943  036232                          TST274:
8944  036232  012700  077777                  MOV     #77777,%0       ;LOAD HIGH ORDER WITH 77777
8945  036236  012701  177777                  MOV     #177777,%0+1    ;LOAD LOW ORDER WITH 177777
8946  036242  071027  000002                  DIV     #2,%0           ;DIVIDE BY #2
8947  036246  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8948  036252  042737  000014  025274          BIC     #14,@#PSWORD
8949  036260  122737  000002  025274          CMPB    #2,@#PSWORD     ;IS PS = 2
8950  036266  001401                          BEQ     13$
8951  036270  104000                          EMT                     ;PS IS WRONG
8952  036272                          13$:
8953  036272  005215                          INC     (R5)
8954
8955  036274  012702  000002                  MOV     #2,%2
8956  036300  012703  025332                  MOV     #S9,%3
8957  036304  012704  025334                  MOV     #S10,%4
8958
8959                                  ;*****************************************************************************
8960                                  ;TEST:275       DIV     0 52525 / S9 = 25252        REM = 1       PS = 0
8961                                  ;*****************************************************************************
8962
8963  036310                          TST275:
8964  036310  012700  000000                  MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
8965  036314  012701  052525                  MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
8966  036320  071067  167006                  DIV     S9,%0           ;DIVIDE BY S9
8967  036324  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
8968  036330  122737  000000  025274          CMPB    #0,@#PSWORD     ;IS PS = 0
8969  036336  001401                          BEQ     11$
8970  036340  104000                          EMT                     ;PS IS WRONG
8971  036342  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
8972  036346  001401                          BEQ     12$
8973  036350  104000                          EMT                     ;QUOTIENT IS WRONG
8974  036352  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
8975  036356  001401                          BEQ     13$
8976  036360  104000                          EMT                     ;WRONG REMAINDER
8977  036362                          13$:
8978  036362  005215                          INC     (R5)
8979
8980                                  ;*****************************************************************************
8981                                  ;TEST:276       DIV     0 52525 / @S10 = 25252        REM = 1       PS = 0
8982                                  ;*****************************************************************************
8983
```

```
 8984  036364
 8985  036364  012700  000000                TST276:
 8986  036370  012701  052525                        MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
 8987  036374  071077  166734                        MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
 8988  036400  106737  025274                        DIV     @S10,%0         ;DIVIDE BY @S10
 8989  036404  122737  000000  025274              MFPS    @#PSWORD        ;SAVE PS
 8990  036412  001401                              CMPB    #0,@#PSWORD     ;IS PS = 0
 8991  036414  104000                              BEQ     11$
 8992  036416  022700  025252                      EMT                     ;PS IS WRONG
 8993  036422  001401                      11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
 8994  036424  104000                              BEQ     12$
 8995  036426  022701  000001                      EMT                     ;QUOTIENT IS WRONG
 8996  036432  001401                      12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
 8997  036434  104000                              BEQ     13$
 8998  036436                                      EMT                     ;WRONG REMAINDER
 8999  036436  005215                      13$:
 9000                                              INC     (R5)
 9001                                  ;*****************************************************************************
 9002                                  ;TEST:277       DIV     0 52525 / @#S9 = 25252       REM = 1       PS = 0
 9003                                  ;*****************************************************************************
 9004
 9005  036440                TST277:
 9006  036440  012700  000000                MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
 9007  036444  012701  052525                MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
 9008  036450  071037  025332                DIV     @#S9,%0         ;DIVIDE BY @#S9
 9009  036454  106737  025274                MFPS    @#PSWORD        ;SAVE PS
 9010  036460  122737  000000  025274        CMPB    #0,@#PSWORD     ;IS PS = 0
 9011  036466  001401                        BEQ     11$
 9012  036470  104000                        EMT                     ;PS IS WRONG
 9013  036472  022700  025252        11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
 9014  036476  001401                        BEQ     12$
 9015  036500  104000                        EMT                     ;QUOTIENT IS WRONG
 9016  036502  022701  000001        12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
 9017  036506  001401                        BEQ     13$
 9018  036510  104000                        EMT                     ;WRONG REMAINDER
 9019  036512                        13$:
 9020  036512  005215                        INC     (R5)
 9021
 9022                                  ;*****************************************************************************
 9023                                  ;TEST:300       DIV     0 52525 / %2 = 25252       REM = 1       PS = 0
 9024                                  ;*****************************************************************************
 9025
 9026  036514                TST300:
 9027  036514  012700  000000                MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
 9028  036520  012701  052525                MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
 9029  036524  071002                        DIV     %2,%0           ;DIVIDE BY %2
 9030  036526  106737  025274                MFPS    @#PSWORD        ;SAVE PS
 9031  036532  122737  000000  025274        CMPB    #0,@#PSWORD     ;IS PS = 0
 9032  036540  001401                        BEQ     11$
 9033  036542  104000                        EMT                     ;PS IS WRONG
 9034  036544  022700  025252        11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
 9035  036550  001401                        BEQ     12$
 9036  036552  104000                        EMT                     ;QUOTIENT IS WRONG
 9037  036554  022701  000001        12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
 9038  036560  001401                        BEQ     13$
 9039  036562  104000                        EMT                     ;WRONG REMAINDER
```

```
9040  036564
9041  036564  005215                          13$:
                                                      INC     (R5)
9042
9043                                          ;*****************************************************************************
9044                                          ;TEST:301      DIV     0 52525 / (3)+ = 25252       REM = 1      PS = 0
9045                                          ;*****************************************************************************
9046
9047  036566                                  TST301:
9048  036566  012700  000000                          MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
9049  036572  012701  052525                          MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
9050  036576  071023                                  DIV     (3)+,%0         ;DIVIDE BY (3)+
9051  036600  106737  025274                          MFPS    @#PSWORD        ;SAVE PS
9052  036604  122737  000000  025274                  CMPB    #0,@#PSWORD     ;IS PS = 0
9053  036612  001401                                  BEQ     11$
9054  036614  104000                                  EMT                     ;PS IS WRONG
9055  036616  022700  025252                  11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
9056  036622  001401                                  BEQ     12$
9057  036624  104000                                  EMT                     ;QUOTIENT IS WRONG
9058  036626  022701  000001                  12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
9059  036632  001401                                  BEQ     13$
9060  036634  104000                                  EMT                     ;WRONG REMAINDER
9061  036636                                  13$:
9062  036636  005215                                  INC     (R5)
9063
9064                                          ;*****************************************************************************
9065                                          ;TEST:302      DIV     0 52525 / -(3) = 25252       REM = 1      PS = 0
9066                                          ;*****************************************************************************
9067
9068  036640                                  TST302:
9069  036640  012700  000000                          MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
9070  036644  012701  052525                          MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
9071  036650  071043                                  DIV     -(3),%0         ;DIVIDE BY -(3)
9072  036652  106737  025274                          MFPS    @#PSWORD        ;SAVE PS
9073  036656  122737  000000  025274                  CMPB    #0,@#PSWORD     ;IS PS = 0
9074  036664  001401                                  BEQ     11$
9075  036666  104000                                  EMT                     ;PS IS WRONG
9076  036670  022700  025252                  11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
9077  036674  001401                                  BEQ     12$
9078  036676  104000                                  EMT                     ;QUOTIENT IS WRONG
9079  036700  022701  000001                  12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
9080  036704  001401                                  BEQ     13$
9081  036706  104000                                  EMT                     ;WRONG REMAINDER
9082  036710                                  13$:
9083  036710  005215                                  INC     (R5)
9084
9085                                          ;*****************************************************************************
9086                                          ;TEST:303      DIV     0 52525 / 2(4) = 25252       REM = 1      PS = 0
9087                                          ;*****************************************************************************
9088
9089  036712                                  TST303:
9090  036712  012700  000000                          MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
9091  036716  012701  052525                          MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
9092  036722  071064  000002                          DIV     2(4),%0         ;DIVIDE BY 2(4)
9093  036726  106737  025274                          MFPS    @#PSWORD        ;SAVE PS
9094  036732  122737  000000  025274                  CMPB    #0,@#PSWORD     ;IS PS = 0
9095  036740  001401                                  BEQ     11$
```

```
9096  036742  104000                          EMT                     ;PS IS WRONG
9097  036744  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
9098  036750  001401                          BEQ     12$
9099  036752  104000                          EMT                     ;QUOTIENT IS WRONG
9100  036754  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
9101  036760  001401                          BEQ     13$
9102  036762  104000                          EMT                     ;WRONG REMAINDER
9103  036764                          13$:
9104  036764  005215                          INC     (R5)
9105
9106                                  ;*****************************************************************************
9107                                  ;TEST:304       DIV     0 52525 / @(4) = 25252       REM = 1      PS = 0
9108                                  ;*****************************************************************************
9109
9110  036766                          TST304:
9111  036766  012700  000000                  MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
9112  036772  012701  052525                  MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
9113  036776  071074  000000                  DIV     @(4),%0         ;DIVIDE BY @(4)
9114  037002  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
9115  037006  122737  000000  025274          CMPB    #0,@#PSWORD     ;IS PS = 0
9116  037014  001401                          BEQ     11$
9117  037016  104000                          EMT                     ;PS IS WRONG
9118  037020  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
9119  037024  001401                          BEQ     12$
9120  037026  104000                          EMT                     ;QUOTIENT IS WRONG
9121  037030  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
9122  037034  001401                          BEQ     13$
9123  037036  104000                          EMT                     ;WRONG REMAINDER
9124  037040                          13$:
9125  037040  005215                          INC     (R5)
9126
9127                                  ;*****************************************************************************
9128                                  ;TEST:305       DIV     0 52525 / @(4)+ = 25252      REM = 1      PS = 0
9129                                  ;*****************************************************************************
9130
9131  037042                          TST305:
9132  037042  012700  000000                  MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
9133  037046  012701  052525                  MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
9134  037052  071034                          DIV     @(4)+,%0        ;DIVIDE BY @(4)+
9135  037054  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
9136  037060  122737  000000  025274          CMPB    #0,@#PSWORD     ;IS PS = 0
9137  037066  001401                          BEQ     11$
9138  037070  104000                          EMT                     ;PS IS WRONG
9139  037072  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
9140  037076  001401                          BEQ     12$
9141  037100  104000                          EMT                     ;QUOTIENT IS WRONG
9142  037102  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
9143  037106  001401                          BEQ     13$
9144  037110  104000                          EMT                     ;WRONG REMAINDER
9145  037112                          13$:
9146  037112  005215                          INC     (R5)
9147
9148                                  ;*****************************************************************************
9149                                  ;TEST:306       DIV     0 52525 / @-(4) = 25252      REM = 1      PS = 0
9150                                  ;*****************************************************************************
9151
```

```
9152  037114
9153  037114  012700  000000          TST306: MOV     #0,%0           ;LOAD HIGH ORDER WITH 0
9154  037120  012701  052525                  MOV     #52525,%0+1     ;LOAD LOW ORDER WITH 52525
9155  037124  071054                          DIV     @-(4),%0        ;DIVIDE BY @-(4)
9156  037126  106737  025274                  MFPS    @#PSWORD        ;SAVE PS
9157  037132  122737  000000  025274          CMPB    #0,@#PSWORD     ;IS PS = 0
9158  037140  001401                          BEQ     11$
9159  037142  104000                          EMT                     ;PS IS WRONG
9160  037144  022700  025252          11$:    CMP     #25252,%0       ;IS QUOTIENT = 25252
9161  037150  001401                          BEQ     12$
9162  037152  104000                          EMT                     ;QUOTIENT IS WRONG
9163  037154  022701  000001          12$:    CMP     #1,%0+1         ;IS REMAINDER = 1
9164  037160  001401                          BEQ     13$
9165  037162  104000                          EMT                     ;WRONG REMAINDER
9166  037164                          13$:
9167  037164  005215                          INC     (R5)
9168
9169
9170
9171                                  ;SPECIAL MULTIPLY DATA PATTERN TEST
9172
9173  037166  012701  177777          TSTSPC: MOV     #-1,R1          ;MAKE R1 -1 SO WE KNOW INSTR. WAS MODIFIER
9174  037172  012700  077700                  MOV     #77700,R0       ;SET UP TEST DATA
9175  037176  070027  000001                  MUL     #1,R0           ;DO MULTIPLY INSTRUCTION
9176  037202  022701  077700                  CMP     #77700,R1       ;CHECK LOW ORDER WORD
9177  037206  001401                          BEQ     1$
9178  037210  104000                          EMT                     ;LOW ORDER PRODUCT ERROR
9179  037212  005700                  1$:     TST     R0              ;CHECK HIGH ORDER WORD
9180  037214  001401                          BEQ     EISEND
9181  037216  104000                          EMT                     ;HIGH ORDER PRODUCT ERROR
9182  037220  000167  000026          EISEND: JMP     MMUTST          ;JMP OVER GARBAGE AND GET TO MMU TEST
```

```
                                .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                                ;*KT11 VECTOR ADDRESS

000250                          MMVEC=  250

                                ;*KT11 STATUS REGISTER ADDRESSES

177572                          SR0=    177572
177574                          SR1=    177574
177576                          SR2=    177576
172516                          SR3=    172516

                                ;*USER "I" PAGE DESCRIPTOR REGISTERS

177600                          UIPDR0= 177600
177602                          UIPDR1= 177602
177604                          UIPDR2= 177604
177606                          UIPDR3= 177606
177610                          UIPDR4= 177610
177612                          UIPDR5= 177612
177614                          UIPDR6= 177614
177616                          UIPDR7= 177616

                                ;*USER "I" PAGE ADDRESS REGISTERS

177640                          UIPAR0= 177640
177642                          UIPAR1= 177642
177644                          UIPAR2= 177644
177646                          UIPAR3= 177646
177650                          UIPAR4= 177650
177652                          UIPAR5= 177652
177654                          UIPAR6= 177654
177656                          UIPAR7= 177656

                                ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

172300                          KIPDR0= 172300
172302                          KIPDR1= 172302
172304                          KIPDR2= 172304
172306                          KIPDR3= 172306
172310                          KIPDR4= 172310
172312                          KIPDR5= 172312
172314                          KIPDR6= 172314
172316                          KIPDR7= 172316

                                ;*KERNEL "I" PAGE ADDRESS REGISTERS

172340                          KIPAR0= 172340
172342                          KIPAR1= 172342
172344                          KIPAR2= 172344
172346                          KIPAR3= 172346
172350                          KIPAR4= 172350
172352                          KIPAR5= 172352
172354                          KIPAR6= 172354
```

```
9239            172356          KIPAR7= 172356
9240
9241            000006          KSP=    SP
9242            000006          USP=    SP
9243            177776          PSW=    PS
9244            000020          TBIT=   20
9245            000100          WBIT=   100
9246            000001          BIT0=   1
9247            000004          ERRVEC= 4
9248
9249                            ;*ADDITIONAL DEFINITIONS
9250                            ;*
9251
9252
9253
9254  037224    000000          WASR6:  .WORD   0       ;USED TO STORE THE STACK POINTER AFTER A TRAP
9255  037226    000000          TRAPPC: .WORD   0       ;USED TO STORE THE PC OF A TRAP OR ABORT
9256  037230    000000          TRAPPS: .WORD   0       ;USED TO STORE THE PS OF A TRAP OR ABORT
9257  037232    000000          WASSR0: .WORD   0       ;USED TO STORE CONTENTS OF SR0
9258  037234    000000          WASSR2: .WORD   0       ;USED TO STORE CONTENTS OR SR2
9259  037236    000000          TBITPS: .WORD   0       ;SAVES THE PSW THAT MAY HAVE ITS T-BIT ON
9260  037240    000000          $TMP0:  .WORD   0       ;TEMPORARY STORAGE LOCATION
9261  037242    000000          $TMP1:  .WORD   0       ;TEMPORARY STORAGE LOCATION
9262  037244    000000          $TMP2:  .WORD   0       ;TEMPORARY STORAGE LOCATION
9263  037246    000000          $TMP3:  .WORD   0       ;TEMPORARY STORAGE LOCATION
9264  037250    000000          $TMP4:  .WORD   0       ;TEMPORARY STORAGE LOCATION
9265
9266
```

```
 9267
 9268  037252  012706  001000          MMUTST: MOV     #STBOT,KSP      ;INITIALIZE THE STACK POINTER
 9269  037256  012767  021236  140764          MOV     #T0250,MMVEC    ;LOAD MEMORY MANAGENT ROUTINE INTO VECTOR
 9270  037264  012767  000340  140760          MOV     #340,MMVEC+2    ;SET NEW PS TO PRIORITY LEVEL 7-KERNEL
 9271  037272  012767  000340  177736          MOV     #340,TBITPS     ;INITIALIZE LOG THAT HOLDS T-BIT PSW
 9272  037300  005067  140266                  CLR     SR0             ;BE SURE MEM. MGMT IS OFF TO START WITH
 9273  037304  032777  000001  161752          BIT     #1,@SWR
 9274  037312  001402                          BEQ     1$
 9275  037314  000167  011010                  JMP     FPSTRT
 9276  037320  012737  050312  000030  1$:     MOV     #ERROR3,@#30    ;SET UP EMT VECTOR TO GO TO RIGHT ERROR CALL
 9277  037326  012737  000002  001004          MOV     #2,@#$TESTN     ;INCREMENT TEST NUMBER
 9278
```

```
                                ;*****************************************************************************
                                ;TEST 351       PSW PRIORITY BIT TEST
                                ;*****************************************************************************
                                TS351:
                                        CLR     R0              ;INITIALIZE R0 WITH PRIORITY=0 DATA
                                2$:     CLR     R1              ;PREPARE R1 TO ACCEPT DATA READ
                                        MTPS    R0              ;WRITE PRIORITY BITS IN THE PSW
                                        MFPS    R1              ;READ BACK THE LOW BYTE OF PSW
                                        BIC     #177437,R1      ;MASK OFF EVERYTHING EXCEPT PRIORITY BITS
                                        CMP     R0,R1           ;WAS CORRECT PRIORITY SET IN THE PSW?
                                        BEQ     3$
                                        EMT                     ;PRIORITY BITS SET WRONG IN PSW
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 2$'' = 000770
                                3$:     ADD     #40,R0          ;CHANGE DATA TO NEXT PRIORITY
                                        CMP     #400,R0         ;HAVE PRIORITIES 0-7 ALL BEEN CHECKED?
                                        BNE     2$              ;BRANCH IF NO

                                ;*****************************************************************************
                                ;TEST 352       PSW MODE BIT TEST
                                ;*****************************************************************************
                                TS352:
                                        CLR     R0              ;INITIALIZE R0 WITH MODE BITS = 0000
                                2$:     CLR     PSW             ;INITIALIZE PSW
                                        BIS     R0,PSW          ;BIT SET THE PSW MODE BITS WITH R0
                                        MOV     PSW,R1          ;READ BACK THE CONTENTS OF THE PSW
                                        BIC     #007777,R1      ;MASK OFF EVERYTHING EXCEPT THE MODE BITS
                                        CMP     R0,R1           ;WERE THE MODE BITS SET CORRECTLY?
                                        BEQ     3$
                                        EMT                     ;MODE BITS SET WRONG IN PSW
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 2$'' = 000763
                                3$:     ADD     #10000,R0       ;CHANGE MODE BIT DATA
                                        BNE     2$              ;BRANCH IF STILL MORE COMBINATIONS
                                        CLR     PSW             ;RESET PSW BEFORE LEAVING

                                ;*****************************************************************************
                                ;TEST 353       BYTE ADDRESSING TEST FOR PSW
                                ;*****************************************************************************
                                TS353:
                                2$:     CLR     PSW             ;CLEAR THE PSW
                                        MOV     #360,R0         ;PUT THE HIGH BYTE DATA INTO R0
                                        MOVB    R0,PSW+1        ;WRITE THE HIGH BYTE OF THE PSW
                                        MOV     PSW,R1          ;READ BACK THE ENTIRE PSW
                                        BIC     #007437,R1      ;MASK OFF THE T & CC BITS
                                        SWAB    R0              ;GET DATA WRITTEN IN HIGH BYTE OF R0
                                        CMP     R0,R1           ;WAS THE PSW WRITTEN TO CORRECTLY
                                        BEQ     4$
                                        EMT                     ;LOW BYTE EFFECTED BY WRITE TO HIGH BYTE OF PSW
                                                                ;FOR TIGHTER SCOPE LOOP
                                                                ;REPLACE ERROR CALL WITH
                                                                ;'BR 2$'' = 000760
```

```
9335  037466  005067  140304          4$:     CLR     PSW             ;CLEAR THE PSW
9336  037472  012700  000340                  MOV     #340,R0         ;PUT THE LOW BYTE DATA INTO R0
9337  037476  110067  140274                  MOVB    R0,PSW          ;WRITE THE LOW BYTE OF THE PSW
9338  037502  016701  140270                  MOV     PSW,R1          ;READ BACK THE ENTIRE PSW
9339  037506  042701  007437                  BIC     #007437,R1      ;MASK OFF THE T&CC BITS
9340  037512  020001                          CMP     R0,R1           ;WAS PSW WRITTEN TO CORRECTLY
9341  037514  001401                          BEQ     TS354
9342  037516  104000                          EMT                     ;HIGH BYTE EFFECTED BY WRITE TO LOW BYTE OF PSW
9343                                                                  ;FOR TIGHTER SCOPE LOOP
9344                                                                  ;REPLACE ERROR CALL WITH
9345                                                                  ;'BR 2$'' = 000736
9346
9347                                  ;*****************************************************************************
9348                                  ;TEST 354       TEST AND SETUP OF STACK POINTERS
9349                                  ;*****************************************************************************
9350  037520                          TS354:
9351  037520  005067  140252                  CLR     PSW             ;GO TO KERNEL MODE
9352  037524  012706  001000                  MOV     #KERSTK,KSP     ;SET KERNEL STACK POINTER TO 1100
9353  037530  012767  140000  140240          MOV     #140000,PSW     ;GO TO USER MODE
9354  037536  012706  000600                  MOV     #USESTK,USP     ;SET USER STACK POINTER TO 700
9355  037542  005067  140230                  CLR     PSW             ;BACK TO KERNEL MODE
9356  037546  022706  001000                  CMP     #KERSTK,KSP     ;IS KERNEL R6 STILL 1100?
9357  037552  001401                          BEQ     TS355
9358  037554  104000                          EMT                     ;KERNEL R6 CHANGED BY WRITING USER R6
9359                                                                  ;FOR TIGHTER SCOPE LOOP
9360                                                                  ;REPLACE ERROR CALL WITH
9361                                                                  ;000756
9362
9363                                  ;*****************************************************************************
9364                                  ;*
9365                                  ;*      THE NEXT FIVE (5) TESTS WILL TRY TO ADDRESS ALL OF THE
9366                                  ;*      MEMORY MANAGEMENT REGISTERS (SR0,SR1,SR2,KERNEL & USER PAR/PDR'S).
9367                                  ;*      EVERY TIME A REGISTER TIMES OUT ITS ADDRESS WILL BE REPORTED.
9368                                  ;*      AT THE END OF EACH TEST A SUMMARY OF THE ADDRESSES THAT TIMED
9369                                  ;*      OUT DURING THAT TEST IS GIVEN.  THE RESULTS OF ''AND-ING'' AND ''OR-ING''
9370                                  ;*      THEIR ADDRESSES IS GIVEN TO SHOW WHICH ADDRESS LINES MAY BE
9371                                  ;*      STUCK AT 0 OR 1.  THE PAR/PDR ADDRESS AND KT MUX'S ARE THE
9372                                  ;*      THINGS BEING CHECKED.
9373                                  ;*
9374                                  ;*****************************************************************************
9375
9376
9377                                  ;*****************************************************************************
9378                                  ;TEST 355       SR0,SR1,SR2,SR3 TIMEOUT TEST
9379                                  ;*****************************************************************************
9380  037556                          TS355:
9381  037556  012700  177572                  MOV     #SR0,R0         ;LOAD R0 WITH ADDRESS OF FIRST REG.
9382  037562  012701  000003                  MOV     #3,R1           ;LOAD R1 WITH THE LOOP COUNT
9383  037566  005710                  2$:     TST     (R0)            ;TRY ADDRESSING A STATUS REGISTER
9384                                                                  ;IF IT TIMES OUT GO TO 5$
9385  037570  062700  000002          3$:     ADD     #2,R0           ;PUT NEXT ADDRESS IN R0
9386  037574  077104                          SOB     R1,2$           ;LOOP BACK TO 2$ UNTIL ALL TESTED
9387  037576  005737  172516                  TST     @#172516        ;CHECK SR3 FOR RESPONSE
9388
9389                                  ;*****************************************************************************
9390                                  ;TEST 356       KERNEL PAR'S TIMEOUT TEST
```

```
                                ;*****************************************************************************
037602                          TS356:
                                
037602  012700  172340                MOV     #KIPAR0,R0      ;LOAD R0 WITH ADDRESS OF FIRST REG.
037606  012701  000010                MOV     #10,R1          ;LOAD R1 WITH LOOP COUNT (8)
037612  005710          2$:     TST     (R0)            ;TRY ADDRESSING A KIPAR
                                                        ;IF IT TIMES OUT, WILL GO TO 5$
037614  062700  000002  3$:     ADD     #2,R0           ;PUT NEXT KIPAR ADDRESS IN R0
037620  077104                        SOB     R1,2$           ;LOOP BACK TO 2$ UNTIL ALL TESTED
                                
                                ;*****************************************************************************
                                ;TEST 357       KERNEL PDR'S TIMEOUT TEST
                                ;*****************************************************************************
037622                          TS357:
                                
037622  012700  172300                MOV     #KIPDR0,R0      ;LOAD R0 WITH ADDRESS OF FIRST REG.
037626  012701  000010                MOV     #10,R1          ;LOAD R1 WITH LOOP COUNT (8)
037632  005710          2$:     TST     (R0)            ;TRY ADDRESSING A KIPDR
                                                        ;IF IT TIMES OUT, WILL GO TO 5$
037634  062700  000002  3$:     ADD     #2,R0           ;PUT NEXT KIPDR ADDRESS IN R0
037640  077104                        SOB     R1,2$           ;LOOP BACK TO 2$ UNTIL ALL TESTED
                                
                                ;*****************************************************************************
                                ;TEST 360       USER PAR'S TIMEOUT TEST
                                ;*****************************************************************************
037642                          TS360:
                                
037642  012700  177640                MOV     #UIPAR0,R0      ;LOAD R0 WITH ADDRESS OF FIRST REG.
037646  012701  000010                MOV     #10,R1          ;LOAD R1 WITH LOOP COUNT (8)
037652  005710          2$:     TST     (R0)            ;TRY ADDRESSING A UIPAR
                                                        ;IF IT TIMES OUT, WILL GO TO 5$
037654  062700  000002  3$:     ADD     #2,R0           ;PUT NEXT UIPAR ADDRESS IN R0
037660  077104                        SOB     R1,2$           ;LOOP BACK TO 2$ UNTIL ALL TESTED
                                
                                ;*****************************************************************************
                                ;TEST 361       USER PDR'S TIMEOUT TEST
                                ;*****************************************************************************
037662                          TS361:
                                
037662  012700  177600                MOV     #UIPDR0,R0      ;LOAD R0 WITH ADDRESS OF FIRST REG.
037666  012701  000010                MOV     #10,R1          ;LOAD R1 WITH LOOP COUNT (8)
037672  005710          2$:     TST     (R0)            ;TRY ADDRESSING A UIPDR
                                                        ;IF IT TIMES OUT, WILL GO TO 5$
037674  062700  000002  3$:     ADD     #2,R0           ;PUT NEXT UIPDR ADDRESS IN R0
037700  077104                        SOB     R1,2$           ;LOOP BACK TO 2$ UNTIL ALL TESTED
                                
                                ;*****************************************************************************
                                ;TEST 362       SR0(15:13) BIT TEST & SR2 TEST
                                ;*****************************************************************************
037702                          TS362:
                                
037702  012700  177572  1$:     MOV     #SR0,R0         ;LOAD ADDRESS OF SR0 INTO R0
037706  012710  160000                MOV     #160000,(R0)    ;SET BITS <15:13> IN SR0 (ERROR BITS)
037712  000005                        RESET                   ;ISSUE AND "INIT" SIGNAL
037714  011001                        MOV     (R0),R1         ;READ SR0 INTO R1 TO SEE IF CLEAR
037716  001401                        BEQ     2$
```

```
9447  037720  104000                          EMT                     ;SR0<15:13> NOT CLEARED BY A "RESET"
9448                                                                    ;FOR TIGHTER SCOPE LOOP
9449                                                                    ;REPLACE ERROR CALL WITH
9450                                                                    ;"BR 1$" = 000770
9451  037722  016767  137650  177304  2$:     MOV     SR2,WASSR2      ;READ CONTENTS OF SR2
9452  037730  012701  037722                  MOV     #2$,R1          ;LOAD EXPECTED CONTENTS INTO R1
9453  037734  020167  177274                  CMP     R1,WASSR2       ;IS SR2 TRACKING?
9454  037740  001401                          BEQ     3$
9455  037742  104000                          EMT                     ;SR2 NOT "TRACKING" VIRTUAL ADDRESSES
9456                                                                    ;FOR TIGHTER SCOPE LOOP
9457                                                                    ;REPLACE ERROR CALL WITH
9458                                                                    ;"BR 2$" = 000767
9459  037744  012701  100000          3$:     MOV     #100000,R1      ;PUT DATA TO BE WRITTEN IN R1
9460  037750  012703  000003                  MOV     #3,R3           ;SETUP R3 AS A LOOP COUNTER
9461  037754  005010                  4$:     CLR     (R0)            ;CLEAR SR0
9462  037756  050110                  5$:     BIS     R1,(R0)         ;SET ONE OF THE ERROR BITS IN SR0
9463  037760  011002                          MOV     (R0),R2         ;READ SR0 INTO R2
9464  037762  020102                          CMP     R1,R2           ;DID RIGHT ERROR BIT GET SET?
9465  037764  001401                          BEQ     6$
9466  037766  104000                          EMT                     ;BITS WERE SET WRONG IN SR0
9467                                                                    ;FOR TIGHTER SCOPE LOOP
9468                                                                    ;REPLACE ERROR CALL WITH
9469                                                                    ;"BR 4$" = 000772
9470  037770  012704  037756          6$:     MOV     #5$,R4          ;LOAD EXPECTED CONTENTS OF SR2 IN R4
9471  037774  016767  137576  177232          MOV     SR2,WASSR2      ;READ SR2
9472  040002  020467  177226                  CMP     R4,WASSR2       ;DID SR2 LOCK UP WHEN ERROR
9473                                                                    ;BIT SET IN SR1?
9474  040006  001401                          BEQ     7$
9475  040010  104000                          EMT                     ;SR2 DID NOT LOCK UP
9476                                                                    ;FOR TIGHTER SCOPE LOOP
9477                                                                    ;REPLACE ERROR CALL WITH
9478                                                                    ;"BR 4$" = 000761
9479  040012  006001          7$:     ROR     R1                      ;CHANGE DATA TO CHECK NEXT ERROR BIT
9480  040014  077321                  SOB     R3,4$                   ;LOOP BACK UNTIL <15:13> ALL TESTED
9481  040016  005010                  CLR     (R0)                    ;CLEAR SR0 BEFORE LEAVING
9482
9483                    ;*****************************************************************************************
9484                    ;TEST 363       SR0 & PSW DUAL ADDRESSING TEST
9485                    ;*****************************************************************************************
9486  040020           TS363:
9487
9488  040020  005067  137752  1$:     CLR     PSW                     ;CLEAR THE PSW
9489  040024  005067  137542          CLR     SR0                     ;CLEAR STATUS REGISTER 0
9490  040030  106427  000340          MTPS    #340                    ;SET PRIORITY 7 IN LOW BYTE OF PSW
9491  040034  016700  137532          MOV     SR0,R0                  ;READ STATUS REGISTER 0
9492  040040  001401                  BEQ     2$
9493  040042  104000                  EMT                             ;SR0 EFFECTED BY A WRITE TO THE PSW
9494                                                                    ;FOR TIGHTER SCOPE LOOP
9495                                                                    ;REPLACE ERROR CALL WITH
9496                                                                    ;"BR 1$" = 000767
9497  040044  005067  137522  2$:     CLR     SR0                     ;BE SURE SR0 IS 0 BEFORE LEAVING
9498  040050  005067  137722          CLR     PSW                     ;BE SURE PSW IS 0 BEFORE LEAVING
9499
9500                    ;*****************************************************************************************
9501                    ;TEST 364       TEST THAT SR1 READS ALL ZEROS
9502                    ;*****************************************************************************************
```

```
 9503  040054
 9504  040054  012700  177777          TS364:
 9505  040060  016700  137510          1$:     MOV     #-1,R0          ;FILL R0 WITH ALL ONES
 9506  040064  001401                          MOV     SR1,R0          ;READ SR1 INTO R0
 9507  040066  104000                          BEQ     2$
 9508                                          EMT                     ;SR1 DID NOT READ ALL ZEROS
 9509                                                                  ;FOR TIGHTER SCOPE LOOP
 9510                                                                  ;REPLACE ERROR CALL WITH
 9511  040070  012767  177777  132420  2$:     MOV     #-1,SR3         ;000772
 9512  040076  022767  000060  132412          CMP     #60,SR3         ;TRY TO WRITE ONES TO SR3
 9513  040104  001401                          BEQ     3$              ;ONLY BITS <5:4> SHOULD BE ONES
 9514  040106  104000                          EMT                     ;DIDN'T READ BACK A ''60''
 9515  040110  004567  010152          3$:     JSR     R5,CHKAPT
 9516  040114  000402                          BR      90$
 9517  040116  000005                          RESET                   ;CLEARS SR3
 9518  040120  000402                          BR      91$
 9519  040122  005067  132370          90$:    CLR     SR3
 9520  040126  005767  132364          91$:    TST     SR3             ;VERIFY THAT IT WAS CLEARED
 9521  040132                          4$:
 9522  040132  001401                          BEQ     TS365
 9523  040134  104000                          EMT                     ;SR3 DIDN'T READ ALL ZEROS
 9524
 9525
 9526                                  ;NOTE   F11 CHANGES INCLUDED CHECKING ALL BITS<15:0> OF PARS
 9527                                  ;       INSTEAD OF ONLY BITS<11:0>.
 9528
 9529                                  ;*******************************************************************************
 9530                                  ;TEST 365       BIT TEST OF KERNEL & USER PAR'S
 9531                                  ;*******************************************************************************
 9532  040136                          TS365:
 9533
 9534  040136  012700  172340          1$:     MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST PAR IN R0
 9535  040142  012703  000010          2$:     MOV     #10,R3          ;SETUP R3 TO COUNT 8 PAR'S
 9536  040146  005010                  3$:     CLR     (R0)            ;CLEAR THE PAR
 9537  040150  011001                          MOV     (R0),R1         ;READ THE PAR INTO R1
 9538  040152  001401                          BEQ     4$
 9539  040154  104000                          EMT                     ;PAR WOULD NOT CLEAR
 9540                                                                  ;FOR TIGHTER SCOPE LOOP
 9541                                                                  ;REPLACE ERROR CALL WITH
 9542                                                                  ;'BR 3$'' = 000774
 9543  040156  012704  077777          4$:     MOV     #077777,R4      ;LOAD 'WALKING 0'' TEST PATTERN IN R4
 9544  040162  005010                  5$:     CLR     (R0)            ;CLEAR THE PAR BEFORE LOADING DATA
 9545  040164  050410                          BIS     R4,(R0)         ;BIT SET THE TEST PATTERN INTO THE PAR
 9546  040166  011002                          MOV     (R0),R2         ;READ THE PAR INTO R2
 9547  040170  020402                          CMP     R4,R2           ;DOES DATA WRITTEN=DATA READ?
 9548  040172  001401                          BEQ     6$
 9549  040174  104000                          EMT                     ;PAR BITS DID NOT SET CORRECTLY
 9550                                                                  ;FOR TIGHTER SCOPE LOOP
 9551                                                                  ;REPLACE ERROR CALL WITH
 9552                                                                  ;'BR 5$'' = 000767
 9553  040176  000261          6$:     SEC                             ;SET THE C-BIT FOR THE ROTATE INST.
 9554  040200  006004                  ROR     R4                      ;ROTATE THE TEST PATTERN IN R4
 9555  040202  103767                  BCS     5$                      ;BRANCH BACK IF MORE BITS TO TEST
 9556  040204  062700  000002          ADD     #2,R0                   ;GET NEXT PAR ADDRESS IN R0
 9557  040210  077322                  SOB     R3,3$                   ;BRANCH BACK UNTIL ALL PAR'S TESTED
 9558  040212  022700  177660          CMP     #UIPAR7+2,R0            ;HAVE USER PAR'S BEEN TESTED
```

```
                        BHIS    TS366           ;GET TO NEXT TEST
                        MOV     #UIPAR0,R0      ;LOAD FIRST USER PAR ADDR. IN R0
                        BR      2$              ;BRANCH BACK TO TEST USER PAR'S
                                                ;LEAVE TEST WITH BITS <11:1>=1 IN ALL PAR'S
                ;*******************************************************************************
                ;TEST 366       BIT TEST OF KERNEL & USER PDR'S
                ;*******************************************************************************
                TS366:

                1$:     MOV     #KIPDR0,R0      ;LOAD ADDRESS OF FIRST PDR IN R0
                2$:     MOV     #10,R3          ;SETUP R3 TO COUNT 8 PDR'S
                3$:     CLR     (R0)            ;CLEAR THE PDR
                        MOV     (R0),R1         ;READ THE PDR INTO R1
                        BEQ     4$
                        EMT                     ;PDR WOULD NOT CLEAR
                                                ;FOR TIGHTER SCOPE LOOP
                                                ;REPLACE ERROR CALL WITH
                                                ;'BR 3$' = 000774
                4$:     MOV     #077777,R4      ;LOAD 'WALKING '0'' TEST PATTERN IN R4
                5$:     CLR     (R0)            ;CLEAR THE PDR BEFORE LOADING DATA
                        MOV     R4,R1           ;LOAD DATA INTO R1
                        BIC     #100361,R1      ;MASK UNUSED BITS OUT OF THE DATA
                        BIS     R1,(R0)         ;BIT SET THE TEST PATTERN INTO THE PDR
                        MOV     (R0),R2         ;READ THE PDR INTO R2
                        CMP     R1,R2           ;DOES DATA WRITTEN=DATA READ?
                        BEQ     6$
                        EMT                     ;PDR BITS DID NOT SET CORRECTLY
                                                ;FOR TIGHTER SCOPE LOOP
                                                ;REPLACE ERROR CALL WITH
                                                ;'BR 5$' = 000767
                6$:     SEC                     ;SET THE C-BIT FOR THE ROTATE INST.
                        ROR     R4              ;ROTATE THE TEST PATTERN IN R4
                        BCS     5$              ;BRANCH BACK IF MORE BITS TO TEST
                        ADD     #2,R0           ;GET NEXT PDR ADDRESS IN R0
                        SOB     R3,3$           ;BRANCH BACK UNTIL ALL PDR'S TESTED
                        CMP     #UIPDR7+2,R0    ;HAVE USER PDR'S BEEN TESTED?
                        BHIS    TS367           ;GET TO NEXT TEST
                        MOV     #UIPDR0,R0      ;LOAD FIRST USER PDR ADDR. IN R0
                        BR      2$              ;BRANCH BACK TO TEST USER PDR'S
                                                ;LEAVE TEST WITH ALL WRITEABLE BITS IN
                                                ;ALL PDR'S = 1

                ;*******************************************************************************
                ;TEST 367       TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PAR'S
                ;*******************************************************************************
                TS367:

                1$:     MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST PAR INTO R0
                        MOV     #10,R3          ;LOAD LOOP COUNTER TO DO 8 PAR'S
                3$:     MOV     #-1,R1          ;LOAD TEST PATTERN INTO R1
                        CLR     (R0)            ;CLEAR THE PAR
                        MOVB    R1,(R0)         ;WRITE 1'S TO THE LOW BYTE OF THE PAR
                        MOV     (R0),R2         ;READ THE ENTIRE PAR INTO R2
                        BIC     #177400,R1      ;MASK HIGH BYTE & UNUSED BITS OUT OF THE DATA
                        CMP     R1,R2           ;WAS ONLY THE LOW BYTE WRITTEN TO
                        BEQ     5$
```

```
 9615  040356  104000                        EMT                     ;HIGH BYTE EFFECTED BY WRITING LOW BYTE IN PAR
 9616                                                                  ;FOR TIGHTER SCOPE LOOP
 9617                                                                  ;REPLACE ERROR CALL WITH
 9618                                                                  ;'BR 3$'' = 000766
 9619  040360  005010                  5$:     CLR     (R0)            ;CLEAR THE PAR
 9620  040362  012701  177777                  MOV     #-1,R1          ;LOAD TEST, PATTERN INTO R1
 9621  040366  110160  000001                  MOVB    R1,1(R0)        ;WRITE 1'S TO THE HIGH BYTE OF THE PAR
 9622  040372  011002                          MOV     (R0),R2         ;READ THE ENTIRE PAR INTO R2
 9623                                                                  ;F11 CHANGE     WAS #170377
 9624  040374  042701  000377                  BIC     #000377,R1      ;MASK LOW BYTE & UNUSED BITS OUT OF DATA
 9625  040400  020102                          CMP     R1,R2           ;WAS ONLY THE HIGH BYTE WRITTEN TO?
 9626  040402  001401                          BEQ     6$
 9627  040404  104000                          EMT                     ;LOW BYTE EFFECTED BY WRITING HIGH BYTE IN PAR
 9628                                                                  ;FOR TIGHTER SCOPE LOOP
 9629                                                                  ;REPLACE ERROR CALL WITH
 9630                                                                  ;'BR 5'' = 000765
 9631  040406  062700  000002          6$:     ADD     #2,R0           ;PUT ADDRESS OF NEXT PAR IN R0
 9632  040412  077330                          SOB     R3,3$           ;BRANCH BACK UNTIL 8 PAR'S TESTED
 9633  040414  022700  177660                  CMP     #UIPAR7+2,R0    ;HAVE USER PAR'S BEEN TESTED
 9634  040420  103003                          BHIS    TS370           ;GET TO NEXT TEST
 9635  040422  012700  177640                  MOV     #UIPAR0,R0      ;LOAD ADDRESS OF FIRST USER PAR IN R0
 9636  040426  000742                          BR      3$              ;BRANCH BACK TO TEST USER PAR'S
 9637
 9638                          ;*****************************************************************************
 9639                          ;TEST 370       TEST FOR DUAL BYTE ADDRESSING OF KERNEL & USER PDR'S
 9640                          ;*****************************************************************************
 9641  040430                  TS370:
 9642
 9643  040430  012700  172300  1$:     MOV     #KIPDR0,R0      ;LOAD ADDRESS OF FIRST PDR INTO R0
 9644  040434  012703  000010          MOV     #10,R3          ;LOAD LOOP COUNTER TO DO 8 PDR'S
 9645  040440  012701  177777  3$:     MOV     #-1,R1          ;LOAD TEST PATTERN INTO R1
 9646  040444  005010                  CLR     (R0)            ;CLEAR THE PDR
 9647  040446  110110                  MOVB    R1,(R0)         ;WRITE 1'S TO THE LOW BYTE OF THE PDR
 9648  040450  011002                  MOV     (R0),R2         ;READ THE ENTIRE PDR INTO R2
 9649  040452  042701  177761          BIC     #177761,R1      ;MASK HIGH BYTE & UNUSED BITS OUT OF DATA
 9650  040456  020102                  CMP     R1,R2           ;WAS ONLY THE LOW BYTE WRITTEN TO?
 9651  040460  001401                  BEQ     5$
 9652  040462  104000                  EMT                     ;HIGH BYTE EFFECTED BY WRITING LOW BYTE IN PDR
 9653                                                          ;FOR TIGHTER SCOPE LOOP
 9654                                                          ;REPLACE ERROR CALL WITH
 9655                                                          ;'BR 3$'' = 000766
 9656  040464  005010          5$:     CLR     (R0)            ;CLEAR THE PDR
 9657  040466  012701  177777          MOV     #-1,R1          ;LOAD TEST PATTERN INTO R1
 9658  040472  110160  000001          MOVB    R1,1(R0)        ;WRITE 1'S TO THE HIGH BYTE OF THE PDR
 9659  040476  011002                  MOV     (R0),R2         ;READ THE ENTIRE PDR INTO R2
 9660  040500  042701  100377          BIC     #100377,R1      ;MASK LOW BYTE & UNUSED BITS OUT OF DATA
 9661  040504  020102                  CMP     R1,R2           ;WAS ONLY THE HIGH BYTE WRITTEN TO?
 9662  040506  001401                  BEQ     6$
 9663  040510  104000                  EMT                     ;LOW BYTE EFFECTED BY WRITING HIGH BYTE IN PDR
 9664                                                          ;FOR TIGHTER SCOPE LOOP
 9665                                                          ;REPLACE ERROR CALL WITH
 9666                                                          ;'BR 5$'' = 000765
 9667  040512  062700  000002  6$:     ADD     #2,R0           ;PUT ADDRESS OF NEXT PDR IN R0
 9668  040516  077330                  SOB     R3,3$           ;BRANCH BACK UNTIL 8 PDR'S TESTED
 9669  040520  022700  177620          CMP     #UIPDR7+2,R0    ;HAVE USER PDR'S BEEN TESTED?
 9670  040524  103003                  BHIS    TS371           ;GET TO NEXT TEST
```

```
9671  040526  012700  177600          MOV     #UIPDR0,R0      ;LOAD ADDRESS OF FIRST USER PDR IN R0
9672  040532  000742                  BR      3$              ;BRANCH BACK TO TEST USER PDR'S
9673
9674                          ;*****************************************************************************
9675                          ;TEST 371       PAR-PDR DUAL ADDRESSING TEST
9676                          ;*****************************************************************************
9677  040534                  TS371:
9678
9679  040534  012703  000010          MOV     #10,R3          ;LOAD LOOP COUNTER WITH AN 8
9680  040540  012700  172300          MOV     #KIPDR0,R0      ;LOAD ADDRESS OF FIRST KERNEL PDR AND R0
9681  040544  004767  007242          JSR     PC,SETREG       ;SET ALL BITS IN ALL PAR'S IN PDR'S
9682  040550  012706  001000  2$:     MOV     #KERSTK,KSP     ;SETUP STACK POINTER
9683  040554  005010                  CLR     (R0)            ;CLEAR ONE OF THE KERNEL PDR'S
9684  040556  004767  007322          JSR     PC,CMPREG       ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
9685  040562  012720  177777          MOV     #-1,(R0)+       ;RESTORE ALL ONES, AND SETUP FOR NEXT PDR
9686  040566  077310                  SOB     R3,2$           ;LOOP TO 2$ UNTIL ALL KERNEL PDR'S CHECKED
9687  040570  012703  000010          MOV     #10,R3          ;LOAD LOOP COUNTER WITH AN 8
9688  040574  012700  172340          MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST KERNEL PAR IN R0
9689  040600  012706  001000  3$:     MOV     #KERSTK,KSP     ;SETUP STACK POINTER
9690  040604  005010                  CLR     (R0)            ;CLEAR ONE OF THE KERNEL PAR'S
9691  040606  004767  007272          JSR     PC,CMPREG       ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
9692  040612  012720  177777          MOV     #-1,(R0)+       ;RESTORE ALL ONES, AND SETUP FOR NEXT PAR
9693  040616  077310                  SOB     R3,3$           ;LOOP TO 3$ UNTIL ALL KERNEL PAR'S CHECKED
9694  040620  012703  000010          MOV     #10,R3          ;LOAD LOOP COUNTER WITH AN 8
9695  040624  012700  177600          MOV     #UIPDR0,R0      ;LOAD ADDRESS OF FIRST USER PDR IN R0
9696  040630  012706  001000  4$:     MOV     #KERSTK,KSP     ;SETUP STACK POINTER
9697  040634  005010                  CLR     (R0)            ;CLEAR ONE OF THE USER PDR'S
9698  040636  004767  007242          JSR     PC,CMPREG       ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
9699  040642  012720  177777          MOV     #-1,(R0)+       ;RESTORE ALL ONES, AND SETUP FOR NEXT UPDR
9700  040646  077310                  SOB     R3,4$           ;LOOP TO 4$ UNTIL ALL USER PDR'S CHECKED
9701  040650  012703  000010          MOV     #10,R3          ;LOAD LOOP COUNTER WITH AN 8
9702  040654  012700  177640          MOV     #UIPAR0,R0      ;LOAD ADDRESS OF FIRST USER PAR IN R0
9703  040660  012706  001000  5$:     MOV     #KERSTK,KSP     ;SETUP STACK POINTER
9704  040664  005010                  CLR     (R0)            ;CLEAR ONE OF THE USER PAR'S
9705  040666  004767  007212          JSR     PC,CMPREG       ;SEE IF OTHER PAR/PDR'S WERE EFFECTED
9706  040672  012720  177777          MOV     #-1,(R0)+       ;RESTORE ALL ONES, AND SETUP FOR NEXT UPAR
9707  040676  077310                  SOB     R3,5$           ;LOOP TO 5$ UNTIL ALL USER PAR'S CHECKED
9708
9709                          ;*****************************************************************************
9710                          ;TEST 372       TEST THAT PAR-PDR'S NOT AFFECTED BY RESET
9711                          ;*****************************************************************************
9712  040700                  TS372:
9713
9714
9715  040700  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
9716  040706  001403                  BEQ     70$             ;IF NO THEN DO TEST
9717  040710  005737  001006          TST     @#$PASS         ;IS THIS FIRST PASS
9718  040714  001063                  BNE     TS373           ;IF NO THEN SHIP TO NEXT TEST
9719  040716                  70$:
9720  040716  004767  007070  1$:     JSR     PC,SETREG       ;SET ALL BITS IN ALL PAR'S AND PDR'S
9721  040722  000005                  RESET                   ;ISSUE AN "INIT" BY EXECUTING A RESET
9722  040724  012700  172300  10$:    MOV     #KIPDR0,R0      ;LOAD ADDRESS OF FIRST KERNEL PDR IN R0
9723  040730  012704  000010          MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
9724  040734  011001  2$:     MOV     (R0),R1         ;READ A KERNEL PDR INTO R1
9725  040736  022701  077416          CMP     #77416,R1       ;ARE ALL THE BITS STILL SET?
9726  040742  001401                  BEQ     3$
```

```
9727  040744  104000                        EMT                     ;KERNEL PDR AFFECTED BY A RESET
9728                                                                  ;FOR TIGHTER SCOPE LOOP
9729                                                                  ;REPLACE ERROR CALL WITH
9730                                                                  ;'BR 2$'' = 000773
9731  040746  062700  000002        3$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT KERNEL PDR
9732  040752  077410                        SOB     R4,2$           ;LOOP TO 2$ UNTIL ALL KERNEL PDR'S CHECKED
9733  040754  012700  172340                MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST KERNEL PAR IN R0
9734  040760  012704  000010                MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
9735  040764  011001                4$:     MOV     (R0),R1         ;READ A KERNEL PAR INTO R1
9736                                                                  ;****F11 CHANGE****     WAS #7777
9737  040766  022701  177777                CMP     #177777,R1      ;ARE ALL THE BITS STILL SET?
9738  040772  001401                        BEQ     5$
9739  040774  104000                        EMT                     ;KERNEL PAR AFFECTED BY A RESET
9740                                                                  ;FOR TIGHTER SCOPE LOOP
9741                                                                  ;REPLACE ERROR CALL WITH
9742                                                                  ;'BR 4$'' = 000773
9743  040776  062700  000002        5$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT KERNEL PAR
9744  041002  077410                        SOB     R4,4$           ;LOOP TO 4$ UNTIL ALL KERNEL PAR'S CHECKED
9745  041004  012700  177600                MOV     #UIPDR0,R0      ;LOAD ADDRESS OF FIRST USER PDR IN R0
9746  041010  012704  000010                MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
9747  041014  011001                6$:     MOV     (R0),R1         ;READ A USER PDR INTO R1
9748  041016  022701  077416                CMP     #77416,R1       ;ARE ALL THE BITS STILL SET?
9749  041022  001401                        BEQ     7$
9750  041024  104000                        EMT                     ;USER PDR AFFECTED BY A RESET
9751                                                                  ;FOR TIGHTER SCOPE LOOP
9752                                                                  ;REPLACE ERROR CALL WITH
9753                                                                  ;'BR 6$'' = 000773
9754  041026  062700  000002        7$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT USER PDR
9755  041032  077410                        SOB     R4,6$           ;LOOP TO 6$ UNTIL ALL USER PDR'S CHECKED
9756
9757  041034  012700  177640                MOV     #UIPAR0,R0      ;LOAD ADDRESS OF FIRST USER PAR IN R0
9758  041040  012704  000010                MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
9759  041044  011001                8$:     MOV     (R0),R1         ;READ A USER PAR INTO R1
9760                                                                  ;****F11 CHANGE****     WAS #7777
9761  041046  022701  177777                CMP     #177777,R1      ;ARE ALL THE BITS STILL SET?
9762  041052  001401                        BEQ     9$
9763  041054  104000                        EMT                     ;USER PAR AFFECTED BY A RESET
9764                                                                  ;FOR TIGHTER SCOPE LOOP
9765                                                                  ;REPLACE ERROR CALL WITH
9766                                                                  ;'BR 8$'' = 000773
9767  041056  062700  000002        9$:     ADD     #2,R0           ;FORM ADDRESS OF NEXT USER PAR
9768  041062  077410                        SOB     R4,8$           ;LOOP TO 8$ UNTIL ALL USER PAR'S CHECKED
9769
9770
9771
9772
9773
9774
9775
9776
9777
9778
9779
9780
9781
9782
```

```
                                ;*****************************************************************************
                                ;TEST 373       RELOCATION & ADDER TEST (NO CARRIES)
                                ;*****************************************************************************
041064                          TS373:

041064  012700  172340        1$:     MOV     #KIPAR0,R0      ;LOAD ADDRESS OF FIRST KERNEL PAR IN R0
041070  005001                        CLR     R1              ;CLEAR R1
041072  012702  000007                MOV     #7,R2           ;LOAD LOOP COUNTER WITH A 7
041076  010120                2$:     MOV     R1,(R0)+        ;MAP KERNEL PAR'S TO PAGES 0-6 (4K EACH)
041100  062701  000200                ADD     #200,R1
041104  077204                        SOB     R2,2$           ;LOOP UNTIL KIPAR0 - KIPAR6 ARE LOADED
041106  012710  177600                MOV     #177600,(R0)    ;MAP KIPAR7 TO THE I/O PAGE
041112  012700  172300                MOV     #KIPDR0,R0      ;LOAD ADDRESS OF FIRST KERNEL PDR IN R0
041116  012701  077406                MOV     #77406,R1       ;LOAD PDR DATA INTO R1
041122  012702  000010                MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
041126  010120                3$:     MOV     R1,(R0)+        ;MAP ALL 8 PAGES 128 BLOCKS, UPWARD
041130  077202                        SOB     R2,3$           ;       EXPANDABLE, READ/WRITE

041132  012700  067776        4$:     MOV     #67776,R0       ;LOAD PHYSICAL ADDR. PBA INTO R0
041136  012701  107776                MOV     #107776,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
041142  012702  125250                MOV     #125250,R2      ;LOAD TEST PATTERN INTO R2
041146  012704  000600                MOV     #600,R4 ;LOAD R4 WITH PAR VALUE
041152  010467  131172                MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
041156  011067  176056                MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
041162  005067  131330                CLR     SR3             ;SET UP FOR 18-BIT ADDRESSING
041166  052767  000001  136376        BIS     #BIT0,SR0       ;TURN ON 'RELOCATION'
041174  010211                        MOV     R2,(R1)         ;LOAD 125250 USING ADDER  (PAR4 + VIRT ADDR.)
041176  005067  136370                CLR     SR0             ;TURN OFF MEMORY MGMT.
041202  011003                        MOV     (R0),R3         ;READ 125250 BACK WITHOUT USING MEM. MGMT.
041204  016710  176030                MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
041210  020203                        CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
                                                            ;WRITTEN USING 'DEST-ONLY-RELOC.'?
041212  001401                        BEQ     5$
041214  104000                        EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
                                                            ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
                                                            ;APPARENTLY PHYSICAL ADDR. WAS
                                                            ;FORMED WRONG BY ADDERS USING
                                                            ;THE VIRTUAL ADDR. AND KIPAR4
                                                            ;FOR TIGHTER SCOPE LOOP
                                                            ;REPLACE ERROR CALL WITH
                                                            ;'BR 4$' = 000742
041216                          5$:
041216  012700  067776        6$:     MOV     #67776,R0       ;LOAD PHYSICAL ADDR. PBA INTO R0
041222  012701  102576                MOV     #102576,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
041226  012702  125251                MOV     #125251,R2      ;LOAD TEST PATTERN INTO R2
041232  012704  000652                MOV     #652,R4 ;LOAD R4 WITH PAR VALUE
041236  010467  131106                MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
```

```
9839  041242  011067  175772          MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
9840  041246  005067  131244          CLR     SR3             ;SET UP FOR 18-BIT ADDRESSING
9841  041252  052767  000001  136312  BIS     #BIT0,SR0       ;TURN ON 'RELOCATION'
9842  041260  010211                  MOV     R2,(R1)         ;LOAD 125251 USING ADDER  (PAR4 + VIRT ADDR.)
9843  041262  005067  136304          CLR     SR0             ;TURN OFF MEMORY MGMT.
9844  041266  011003                  MOV     (R0),R3         ;READ 125251 BACK WITHOUT USING MEM. MGMT.
9845  041270  016710  175744          MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
9846  041274  020203                  CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
9847                                                          ;WRITTEN USING 'DEST-ONLY-RELOC.'?
9848  041276  001401                  BEQ     7$
9849  041300  104000                  EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
9850                                                          ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
9851                                                          ;APPARENTLY PHYSICAL ADDR. WAS
9852                                                          ;FORMED WRONG BY ADDERS USING
9853                                                          ;THE VIRTUAL ADDR. AND KIPAR4
9854                                                          ;FOR TIGHTER SCOPE LOOP
9855                                                          ;REPLACE ERROR CALL WITH
9856                                                          ;'BR 6$' = 000742
9857  041302                  7$:
9858  041302  012700  067776  8$:     MOV     #67776,R0       ;LOAD PHYSICAL ADDR. PBA INTO R0
9859  041306  012701  105276          MOV     #105276,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
9860  041312  012702  125252          MOV     #125252,R2      ;LOAD TEST PATTERN INTO R2
9861  041316  012704  000625          MOV     #625,R4 ;LOAD R4 WITH PAR VALUE
9862  041322  010467  131022          MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
9863  041326  011067  175706          MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
9864  041332  052767  000020  131156  BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
9865  041340  052767  000001  136224  BIS     #BIT0,SR0       ;TURN ON 'RELOCATION'
9866  041346  010211                  MOV     R2,(R1)         ;LOAD 125252 USING ADDER  (PAR4 + VIRT ADDR.)
9867  041350  005067  136216          CLR     SR0             ;TURN OFF MEMORY MGMT.
9868  041354  011003                  MOV     (R0),R3         ;READ 125252 BACK WITHOUT USING MEM. MGMT.
9869  041356  016710  175656          MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
9870  041362  020203                  CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
9871                                                          ;WRITTEN USING 'DEST-ONLY-RELOC.'?
9872  041364  001401                  BEQ     9$
9873  041366  104000                  EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
9874                                                          ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
9875                                                          ;APPARENTLY PHYSICAL ADDR. WAS
9876                                                          ;FORMED WRONG BY ADDERS USING
9877                                                          ;THE VIRTUAL ADDR. AND KIPAR4
9878                                                          ;FOR TIGHTER SCOPE LOOP
9879                                                          ;REPLACE ERROR CALL WITH
9880                                                          ;'BR 8$' = 000742
9881  041370                  9$:
9882
9883  041370  012700  177776          MOV     #PSW,R0         ;LOAD PHYS. ADDR. OF PSW INTO R0
9884  041374  012701  100076          MOV     #100076,R1      ;LOAD VIRTUAL ADDR. FOR PSW INTO R1
9885  041400  012702  030340          MOV     #030340,R2      ;LOAD DATA FOR PSW IN R2
9886  041404  012704  007777          MOV     #7777,R4        ;LOAD R4 WITH PAR VALUE
9887  041410  010467  130734          MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
9888  041414  005010                  CLR     (R0)            ;CLEAR THE PSW
9889  041416  005067  131074          CLR     SR3             ;SET UP FOR 18-BIT ADDRESSING
9890  041422  052767  000001  136142  BIS     #BIT0,SR0       ;TURN ON 'MEMORY MANAGEMENT'
9891  041430  010211                  MOV     R2,(R1)         ;LOAD PSW USING ADDER (PAR4 + VIRT ADDR.)
9892  041432  005067  136134          CLR     SR0             ;TURN OFF MEM. MGMT (SR0=0)
9893  041436  011003                  MOV     (R0),R3         ;READ PSW BACK WITHOUT USING MEM. MGMT.
9894  041440  005010                  CLR     (R0)            ;CLEAR THE PSW
```

```
9895  041442  042703  000037            BIC     #37,R3          ;MASK T-BIT & CC BITS OUT OF DATA READ
9896  041446  020203                    CMP     R2,R3           ;WAS PSW WRITTEN?
9897  041450  001401                    BEQ     11$
9898  041452  104000                    EMT                     ;PSW DID NOT HAVE DATA THAT IT SHOULD HAVE,
9899                                                            ;APPARENTLY PHYS. ADDR. OF PSW WAS
9900                                                            ;NOT FORMED BY ADDERS USING THE
9901                                                            ;VIRTUAL ADDR. AND KIPAR4
9902                                                            ;FOR TIGHTER SCOPE LOOP
9903                                                            ;REPLACE ERROR CALL WITH
9904                                                            ;'BR 10$'' = 000742
9905  041454  012700  177776            11$:    MOV     #PSW,R0         ;LOAD PHYS. ADDR. OF PSW INTO R0
9906  041460  012701  117776                    MOV     #117776,R1      ;LOAD VIRTUAL ADDR. FOR PSW INTO R1
9907  041464  012702  030240                    MOV     #030240,R2      ;LOAD DATA FOR PSW IN R2
9908  041470  012704  177600                    MOV     #177600,R4      ;LOAD R4 WITH PAR VALUE
9909  041474  010467  130650                    MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
9910  041500  052767  000020  131010            BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
9911  041506  052767  000001  136056            BIS     #BIT0,SR0       ;TURN ON 'MEMORY MANAGEMENT''
9912  041514  010211                    MOV     R2,(R1)         ;LOAD PSW USING ADDER (PAR4 + VIRT. ADDR.)
9913  041516  005067  136050            CLR     SR0             ;TURN OFF MEM. MGMT (SR0=0)
9914  041522  011003                    MOV     (R0),R3         ;READ PSW BACK WITHOUT USING MEM. MGMT.
9915  041524  005010                    CLR     (R0)            ;CLEAR THE PSW
9916  041526  042703  000037            BIC     #37,R3          ;MASK T-BIT & CC BITS OUT OF DATA READ
9917  041532  020203                    CMP     R2,R3           ;WAS PSW WRITTEN WHILE IN MAINT. MODE?
9918  041534  001401                    BEQ     TS374
9919  041536  104000                    EMT                     ;PSW DID NOT HAVE DATA THAT IT SHOULD
9920                                                            ;HAVE, APPARENTLY PHYS. ADDR. OF PSW WAS
9921                                                            ;NOT FORMED BY ADDERS USING THE
9922                                                            ;VIRTUAL ADDR. AND KIPAR4
9923                                                            ;FOR TIGHTER SCOPE LOOP
9924                                                            ;REPLACE ERROR CALL WITH
9925                                                            ;'BR 11$'' = 000743
9926
9927                                ;*****************************************************************************************
9928                                ;TEST 374       RELOCATION & ADDER TEST (WITH CARRIES)
9929                                ;*****************************************************************************************
9930  041540                        TS374:
9931
9932  041540                        1$:                             ;KERNEL PAR'S AND PDR'S HAVE BEEN
9933                                                                ;SETUP BY THE PREVIOUS TEST
9934  041540  012700  066476        2$:     MOV     #66476,R0       ;LOAD PHYSICAL ADDR. PBA INTO R0
9935  041544  012701  114376                MOV     #114376,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
9936  041550  012702  125253                MOV     #125253,R2      ;LOAD TEST PATTERN INTO R2
9937  041554  012704  000521                MOV     #521,R4 ;LOAD R4 WITH PAR VALUE
9938  041560  010467  130564                MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
9939  041564  011067  175450                MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
9940  041570  052767  000020  130720        BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
9941  041576  052767  000001  135766        BIS     #BIT0,SR0       ;TURN ON 'RELOCATION''
9942  041604  010211                    MOV     R2,(R1)         ;LOAD 125253 USING ADDER  (PAR4 + VIRT ADDR.)
9943  041606  005067  135760            CLR     SR0             ;TURN OFF MEMORY MGMT.
9944  041612  011003                    MOV     (R0),R3         ;READ 125253 BACK WITHOUT USING MEM. MGMT.
9945  041614  016710  175420            MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
9946  041620  020203                    CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
9947                                                            ;WRITTEN USING 'DEST-ONLY-RELOC.''?
9948  041622  001401                    BEQ     3$
9949  041624  104000                    EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
9950                                                            ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
```

```
 9951                                                   ;APPARENTLY PHYSICAL ADDR. WAS
 9952                                                   ;FORMED WRONG BY ADDERS USING
 9953                                                   ;THE VIRTUAL ADDR. AND KIPAR4
 9954                                                   ;FOR TIGHTER SCOPE LOOP
 9955                                                   ;REPLACE ERROR CALL WITH
 9956                                                   ;'BR 2$' = 000742
 9957  041626                          3$:
 9958  041626  012700  062276          4$:    MOV     #62276,R0       ;LOAD PHYSICAL ADDR. PBA INTO R0
 9959  041632  012701  107376                 MOV     #107376,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
 9960  041636  012702  125254                 MOV     #125254,R2      ;LOAD TEST PATTERN INTO R2
 9961  041642  012704  000527                 MOV     #527,R4 ;LOAD R4 WITH PAR VALUE
 9962  041646  010467  130476                 MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
 9963  041652  011067  175362                 MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
 9964  041656  052767  000020  130632         BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
 9965  041664  052767  000001  135700         BIS     #BIT0,SR0       ;TURN ON 'RELOCATION'
 9966  041672  010211                         MOV     R2,(R1)         ;LOAD 125254 USING ADDER  (PAR4 + VIRT ADDR.)
 9967  041674  005067  135672                 CLR     SR0             ;TURN OFF MEMORY MGMT.
 9968  041700  011003                         MOV     (R0),R3         ;READ 125254 BACK WITHOUT USING MEM. MGMT.
 9969  041702  016710  175332                 MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
 9970  041706  020203                         CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
 9971                                                                 ;WRITTEN USING 'DEST-ONLY-RELOC.'?
 9972  041710  001401                         BEQ     5$
 9973  041712  104000                         EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
 9974                                                                 ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
 9975                                                                 ;APPARENTLY PHYSICAL ADDR. WAS
 9976                                                                 ;FORMED WRONG BY ADDERS USING
 9977                                                                 ;THE VIRTUAL ADDR. AND KIPAR4
 9978                                                                 ;FOR TIGHTER SCOPE LOOP
 9979                                                                 ;REPLACE ERROR CALL WITH
 9980                                                                 ;'BR 4$' = 000742
 9981  041714                          5$:
 9982  041714  012700  062076          6$:    MOV     #62076,R0       ;LOAD PHYSICAL ADDR. PBA INTO R0
 9983  041720  012701  104576                 MOV     #104576,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
 9984  041724  012702  125255                 MOV     #125255,R2      ;LOAD TEST PATTERN INTO R2
 9985  041730  012704  000553                 MOV     #553,R4 ;LOAD R4 WITH PAR VALUE
 9986  041734  010467  130410                 MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
 9987  041740  011067  175274                 MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
 9988  041744  052767  000020  130544         BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
 9989  041752  052767  000001  135612         BIS     #BIT0,SR0       ;TURN ON 'RELOCATION'
 9990  041760  010211                         MOV     R2,(R1)         ;LOAD 125255 USING ADDER  (PAR4 + VIRT ADDR.)
 9991  041762  005067  135604                 CLR     SR0             ;TURN OFF MEMORY MGMT.
 9992  041766  011003                         MOV     (R0),R3         ;READ 125255 BACK WITHOUT USING MEM. MGMT.
 9993  041770  016710  175244                 MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
 9994  041774  020203                         CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
 9995                                                                 ;WRITTEN USING 'DEST-ONLY-RELOC.'?
 9996  041776  001401                         BEQ     7$
 9997  042000  104000                         EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
 9998                                                                 ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
 9999                                                                 ;APPARENTLY PHYSICAL ADDR. WAS
10000                                                                 ;FORMED WRONG BY ADDERS USING
10001                                                                 ;THE VIRTUAL ADDR. AND KIPAR4
10002                                                                 ;FOR TIGHTER SCOPE LOOP
10003                                                                 ;REPLACE ERROR CALL WITH
10004                                                                 ;'BR 6$' = 000742
10005  042002                          7$:
10006  042002  012700  000000          8$:    MOV     #000000,R0      ;LOAD PHYSICAL ADDR. PBA INTO R0
```

```
10007  042006  012701  111400                    MOV     #111400,R1      ;LOAD VIRTUAL ADDR. VBA INTO R1
10008  042012  012702  125256                    MOV     #125256,R2      ;LOAD TEST PATTERN INTO R2
10009  042016  012704  177664                    MOV     #177664,R4      ;LOAD R4 WITH PAR VALUE
10010  042022  010467  130322                    MOV     R4,KIPAR4       ;LOAD KERNEL PAR 4 BITS <11:00>
10011  042026  011067  175206                    MOV     (R0),$TMP0      ;SAVE CONTENTS AT TEST LOCATION
10012  042032  052767  000020  130456            BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
10013  042040  052767  000001  135524            BIS     #BIT0,SR0       ;TURN ON "RELOCATION"
10014  042046  010211                            MOV     R2,(R1)         ;LOAD 125256 USING ADDER  (PAR4 + VIRT ADDR.)
10015  042050  005067  135516                    CLR     SR0             ;TURN OFF MEMORY MGMT.
10016  042054  011003                            MOV     (R0),R3         ;READ 125256 BACK WITHOUT USING MEM. MGMT.
10017  042056  016710  175156                    MOV     $TMP0,(R0)      ;RESTORE ORIGINAL CONTENTS TO TEST LOC.
10018  042062  020203                            CMP     R2,R3           ;WAS SAME PATTERN READ BACK THAT WAS
10019                                                                    ;WRITTEN USING "DEST-ONLY-RELOC."?
10020  042064  001401                            BEQ     9$
10021  042066  104000                            EMT                     ;TEST LOCATION DID NOT HAVE PATTERN
10022                                                                    ;THAT SHOULD HAVE BEEN WRITTEN TO IT.
10023                                                                    ;APPARENTLY PHYSICAL ADDR. WAS
10024                                                                    ;FORMED WRONG BY ADDERS USING
10025                                                                    ;THE VIRTUAL ADDR. AND KIPAR4
10026                                                                    ;FOR TIGHTER SCOPE LOOP
10027                                                                    ;REPLACE ERROR CALL WITH
10028                                                                    ;"BR 8$" = 000742
10029  042070                            9$:
10030
10031                                    ;*****************************************************************************
10032                                    ;TEST 375       READ AND WRITE WHILE IN RELOCATE MODE
10033                                    ;*****************************************************************************
10034  042070                            TS375:
10035
10036  042070  005067  135702            1$:     CLR     PSW             ;START IN KERNEL MODE
10037  042074  012704  001377                    MOV     #1377,R4        ;LOAD R4 WITH VALUE FOR PAR4
10038  042100  012705  001400                    MOV     #1400,R5        ;LOAD R5 WITH VALUE FOR PAR5
10039  042104  010467  130240                    MOV     R4,KIPAR4       ;LOAD KERNEL PAR4
10040  042110  010567  130236                    MOV     R5,KIPAR5       ;LOAD KERNEL PAR5
10041  042114  012700  177640                    MOV     #UIPAR0,R0      ;LOAD ADDRESS OF FIRST USER PAR IN R0
10042  042120  005001                            CLR     R1              ;CLEAR R1
10043  042122  012702  000007                    MOV     #7,R2           ;LOAD LOOP COUNTER WITH A 7
10044  042126  010120                    2$:     MOV     R1,(R0)+        ;MAP USER PAR'S TO PAGES 0-6 (4K EACH)
10045  042130  062701  000200                    ADD     #200,R1
10046  042134  077204                            SOB     R2,2$           ;LOOP UNTIL UIPAR0-UIPAR6 ARE LOADED
10047  042136  012710  177600                    MOV     #177600,(R0)    ;MAP USER PAR7 TO THE I/O PAGE
10048  042142  012700  177600                    MOV     #UIPDR0,R0      ;LOAD ADDRESS OF FIRST USER PDR IN R0
10049  042146  012701  077406                    MOV     #77406,R1       ;LOAD PDR DATA INTO R1
10050  042152  012702  000010                    MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
10051  042156  010120                    3$:     MOV     R1,(R0)+        ;MAP ALL 8 PAGES 128 BLOCKS, UPWARD
10052  042160  077202                            SOB     R2,3$           ;       EXPANDABLE, READ/WRITE
10053  042162  012767  042426  136060            MOV     #8$,MMVEC       ;SET M. M. TRAP VECTOR TO 8$
10054  042170  052767  000020  130320            BIS     #BIT4,SR3       ;SET UP FOR 22-BIT ADDRESSING
10055  042176  012767  000001  135366            MOV     #BIT0,SR0       ;TURN ON MEMORY MANAGEMENT
10056  042204  105067  135400                    CLRB    UIPDR4          ;MAP USER SPACE NON-RESIDENT WHILE
10057  042210  105067  135376                    CLRB    UIPDR5          ;       TESTING KERNEL SPACE
10058  042214  010567  135430                    MOV     R5,UIPAR4       ;MAP USER PAR'S OPPOSITE OF KIPAR'S
10059  042220  010467  135426                    MOV     R4,UIPAR5
10060  042224  016767  135546  175006    4$:     MOV     PSW,$TMP0       ;SAVE PSW IN CASE OF ERROR
10061  042232  012700  100100                    MOV     #100100,R0      ;PUT VIRTUAL ADDR. THAT USES PAR4 IN R0
10062  042236  012701  120000                    MOV     #120000,R1      ;PUT VIRTUAL ADDR. THAT USES PAR5 IN R1
```

```
10063  042242  010010                  5$:   MOV   R0,(R0)       ;WRITE TO TEST LOC. USING PAR4
10064  042244  011102                        MOV   (R1),R2       ;READ THE SAME LOC., BUT USING PAR5
10065  042246  020002                        CMP   R0,R2         ;DID WE READ WHAT WE WROTE?
10066  042250  001401                        BEQ   6$
10067  042252  104000                        EMT                 ;READING LOC. USING PAR5 AND A VIRT.
10068                                                             ;ADDR. DID NOT FIND DATA WRITTEN WHEN USING
10069                                                             ;PAR4 AND VIRT. ADDRESS.
10070                                                             ;FOR TIGHTER SCOPE LOOP
10071                                                             ;REPLACE ERROR CALL WITH
10072                                                             ;''BR 5$'' = 000765
10073  042254  062700  000100            6$:   ADD   #100,R0       ;CHANGE VIRTUAL ADDRS. TO POINT TO NEXT BLOCK
10074  042260  062701  000100                  ADD   #100,R1
10075  042264  020127  127700                  CMP   R1,#127700    ;WERE BLOCKS FROM 60000-676000 ALL TRIED?
10076  042270  001364                        BNE   5$            ;BRANCH IF NO
10077  042272  032767  140000  135476        BIT   #140000,PSW   ;HAVE WE DONE TEST IN USER MODE YET?
10078  042300  001026                        BNE   7$            ;BRANCH IF YES
10079  042302  010467  135342              MOV   R4,UIPAR4     ;LOAD USER PAR4
10080  042306  010567  135340              MOV   R5,UIPAR5     ;LOAD USER PAR5
10081  042312  112767  000006  135270      MOVB  #6,UIPDR4     ;MAP USER SPACE R/W TO TEST IT
10082  042320  112767  000006  135264      MOVB  #6,UIPDR5
10083  042326  105067  127756              CLRB  KIPDR4        ;MAP  KERNEL SPACE NON-RESIDENT WHILE
10084  042332  105067  127754              CLRB  KIPDR5        ;       TESTING USER SPACE
10085  042336  010567  130006              MOV   R5,KIPAR4     ;MAP KERNEL PAR'S OPPOSITE UIPAR'S
10086  042342  010467  130004              MOV   R4,KIPAR5
10087  042346  012767  140000  135422      MOV   #140000,PSW   ;GO TO USER MODE
10088  042354  000723                        BR    4$            ;GO BACK AND READ/WRITE IN USER MODE
10089  042356  005067  135414            7$:   CLR   PSW           ;GO BACK TO KERNEL MODE BEFORE LEAVING
10090  042362  012767  077406  127720      MOV   #77406,KIPDR4 ;REMAP KERNEL PAGES READ/WRITE
10091  042370  012767  077406  127714      MOV   #77406,KIPDR5
10092  042376  010567  127746              MOV   R5,KIPAR4     ;MAP KERNEL  AND USER PAR'S 4 & 5
10093  042402  010567  127744              MOV   R5,KIPAR5     ;  BACK TO 12-16K
10094  042406  010567  135236              MOV   R5,UIPAR4
10095  042412  010567  135234              MOV   R5,UIPAR5
10096  042416  012767  021236  135624      MOV   #T0250,MMVEC  ;RESTORE ADDR. OF NORMAL M.M. TRAP ROUTINE
10097  042424  000404                        BR    T5376         ;GET TO NEXT TEST
10098  042426  042767  160000  135136  8$:   BIC   #160000,SR0     ;CLEAR ERROR BITS IN SR0
10099  042434  104000                        EMT                 ;M.M. TRAP WHILE IN RELOCATE MODE -
10100                                                             ;REFERENCED WRONG SET OF PDR'S
10101                                                             ;FOR TIGHTER SCOPE LOOP
10102                                                             ;REPLACE ERROR CALL WITH
10103                                                             ;A ''NOP'' = 000240
10104
10105                                ;*****************************************************************************
10106                                ;TEST 376       W-BIT LOGIC TEST, KERNEL PDR'S
10107                                ;*****************************************************************************
10108  042436                        T5376:
10109  042436                        1$:
10110  042436  004767  005262              JSR   PC,TOFF       ;TURN T-BIT TRAPPING OFF FOR THIS TEST
10111  042442  012702  000004              MOV   #4,R2         ;SET LOOP COUNTER TO 4
10112  042446  012700  172346              MOV   #KIPAR3,R0    ;LOAD ADDRESS OF PAR3 INTO R0
10113  042452  012701  001400              MOV   #1400,R1      ;LOAD ''24-28K'' PAR VALUE INTO R1
10114  042456  010120                  2$:   MOV   R1,(R0)+      ;MAP PARS 3-6 TO 12-16K
10115  042460  077202                        SOB   R2,2$         ;LOOP TIL ALL 4 OF THEM LOADED
10116  042462  012705  172300              MOV   #KIPDR0,R5    ;LOAD ADDRESS OF FIRST PDR TO BE TESTED IN R5
10117  042466  012704  000010              MOV   #10,R4        ;SET LOOP COUNTER TO 8
10118  042472  012703  017776              MOV   #17776,R3     ;INITIALIZE VIRTUAL ADDRESS TO BE IN R3
```

```
10119  042476  012700  172300          3$:     MOV     #KIPDR0,R0      ;LOAD ADDR. OF FIRST PDR TO BE SETUP IN R0
10120  042502  012702  000010                  MOV     #10,R2          ;SET LOOP COUNTER TO 8
10121  042506  012701  077406                  MOV     #77406,R1       ;PUT 'W-BIT OFF DATA' INTO R1
10122  042512  010120                  4$:     MOV     R1,(R0)+        ;CLEAR ALL W-BITS BY WRITING TO ALL PDRS
10123  042514  077202                          SOB     R2,4$           ;LOOP UNTIL ALL OF THEM SETUP
10124  042516  011313                          MOV     (R3),(R3)       ;DO 'DATO' TO VIRTUAL ADDR.-SETTING A W-BIT
10125  042520  031527  000100                  BIT     (R5),#WBIT      ;DID THAT CAUSE W-BIT TO BE SET?
10126  042524  001001                          BNE     5$
10127  042526  104000                          EMT                     ;W-BIT DID NOT GET SET IN PDR
10128                                                                  ;FOR TIGHTER SCOPE LOOP
10129                                                                  ;REPLACE ERROR CALL WITH
10130                                                                  ;'BR 3$' = 000763
10131  042530  012702  000010          5$:     MOV     #10,R2          ;SET LOOP COUNTER TO 8
10132  042534  012700  172300                  MOV     #KIPDR0,R0      ;LOAD ADDR. OF FIRST PDR TO BE CHECKED IN R0
10133  042540  031027  000100          6$:     BIT     (R0),#WBIT      ;DID W-BIT IN OTHER PDRS REMAIN CLEAR?
10134  042544  001403                          BEQ     7$              ;BRANCH IF YES
10135  042546  020500                          CMP     R5,R0           ;IF W-BIT SET, THEN WAS IT PDR UNDER TEST?
10136  042550  001401                          BEQ     7$
10137  042552  104000                          EMT                     ;W-BIT GOT SET IN MORE THAN ONE PDR
10138                                                                  ;FOR TIGHTER SCOPE LOOP
10139                                                                  ;REPLACE ERROR CALL WITH
10140                                                                  ;'BR 3$' = 000750
10141  042554  062700  000002          7$:     ADD     #2,R0           ;POINT R0 TO NEXT PDR TO BE CHECKED
10142  042560  077211                          SOB     R2,6$           ;LOOP UNTIL ALL 8 CHECKED FOR CLEAR W-BIT
10143  042562  010115                          MOV     R1,(R5)         ;WRITE TO THE PDR TESTED TO CLEAR W-BIT
10144  042564  031527  000100                  BIT     (R5),#WBIT      ;DID WRITING PDR CLEAR THE W-BIT?
10145  042570  001401                          BEQ     8$
10146  042572  104000                          EMT                     ;W-BIT DID NOT CLEAR BY WRITING THE PDR
10147                                                                  ;FOR TIGHTER SCOPE LOOP
10148                                                                  ;REPLACE ERROR CALL WITH
10149                                                                  ;'BR 3$' = 000740
10150  042574  062705  000002          8$:     ADD     #2,R5           ;POINT R5 TO THE NEXT PDR TO BE TESTED
10151  042600  062703  020000                  ADD     #20000,R3       ;CHANGE VIRT. ADDR TO REF. NEXT PDR
10152  042604  077444                          SOB     R4,3$           ;LOOP BACK TO 3$ UNTIL ALL 8 PDR'S TESTED
10153  042606  004767  005146                  JSR     PC,TON          ;TURN T-BIT BACK ON FOR NEXT TEST
10154
10155                                  ;*********************************************************************************************
10156                                  ;TEST 377       W-BIT LOGIC TEST, USER PDR'S
10157                                  ;*********************************************************************************************
10158  042612                          TS377:
10159  042612  012767  140000  135156  1$:     MOV     #140000,PSW     ;GO TO USER MODE FOR THIS TEST
10160  042620  004767  005100                  JSR     PC,TCFF         ;TURN T-BIT TRAPPING OFF FOR THIS TEST
10161  042624  012702  000004                  MOV     #4,R2           ;SET LOOP COUNTER TO 4
10162  042630  012700  177646                  MOV     #UIPAR3,R0      ;LOAD ADDRESS OF PAR3 INTO R0
10163  042634  012701  001400                  MOV     #1400,R1        ;LOAD '24-28K' PAR VALUE INTO R1
10164  042640  010120                  2$:     MOV     R1,(R0)+        ;MAP PARS 3-6 TO 12-16K
10165  042642  077202                          SOB     R2,2$           ;LOOP TIL ALL 4 OF THEM LOADED
10166  042644  012705  177600                  MOV     #UIPDR0,R5      ;LOAD ADDRESS OF FIRST PDR TO BE TESTED IN R5
10167  042650  012704  000010                  MOV     #10,R4          ;SET LOOP COUNTER TO 8
10168  042654  012703  017776                  MOV     #17776,R3       ;INITIALIZE VIRTUAL ADDRESS TO BE IN R3
10169  042660  012700  177600          3$:     MOV     #UIPDR0,R0      ;LOAD ADDR. OF FIRST PDR TO BE SETUP IN R0
10170  042664  012702  000010                  MOV     #10,R2          ;SET LOOP COUNTER TO 8
10171  042670  012701  077406                  MOV     #77406,R1       ;PUT 'W-BIT OFF DATA' INTO R1
10172  042674  010120                  4$:     MOV     R1,(R0)+        ;CLEAR ALL W-BITS BY WRITING TO ALL PDRS
10173  042676  077202                          SOB     R2,4$           ;LOOP UNTIL ALL OF THEM SETUP
10174  042700  011313                          MOV     (R3),(R3)       ;DO 'DATO' TO VIRTUAL ADDR.-SETTING A W-BIT
```

```
10175  042702  031527  000100          BIT     (R5),#WBIT      ;DID THAT CAUSE W-BIT TO BE SET?
10176  042706  001001                  BNE     5$
10177  042710  104000                  EMT                     ;W-BIT DID NOT GET SET IN PDR
10178                                                          ;FOR TIGHTER SCOPE LOOP
10179                                                          ;REPLACE ERROR CALL WITH
10180                                                          ;'BR 3$'' = 000763
10181  042712  012702  000010  5$:     MOV     #10,R2          ;SET LOOP COUNTER TO 8
10182  042716  012700  177600          MOV     #UIPDR0,R0      ;LOAD ADDR. OF FIRST PDR TO BE CHECKED IN R0
10183  042722  031027  000100  6$:     BIT     (R0),#WBIT      ;DID W-BIT IN OTHER PDRS REMAIN CLEAR?
10184  042726  001403                  BEQ     7$              ;BRANCH IF YES
10185  042730  020500                  CMP     R5,R0           ;IF W-BIT SET, THEN WAS IT PDR UNDER TEST?
10186  042732  001401                  BEQ     7$
10187  042734  104000                  EMT                     ;W-BIT GOT SET IN MORE THAN ONE PDR
10188                                                          ;FOR TIGHTER SCOPE LOOP
10189                                                          ;REPLACE ERROR CALL WITH
10190                                                          ;'BR 3$'' = 000750
10191  042736  062700  000002  7$:     ADD     #2,R0           ;POINT R0 TO NEXT PDR TO BE CHECKED
10192  042742  077211                  SOB     R2,6$           ;LOOP UNTIL ALL 8 CHECKED FOR CLEAR W-BIT
10193  042744  010115                  MOV     R1,(R5)         ;WRITE TO THE PDR TESTED TO CLEAR W-BIT
10194  042746  031527  000100          BIT     (R5),#WBIT      ;DID WRITING PDR CLEAR THE W-BIT?
10195  042752  001401                  BEQ     8$
10196  042754  104000                  EMT                     ;W-BIT DID NOT CLEAR BY WRITING THE PDR
10197                                                          ;FOR TIGHTER SCOPE LOOP
10198                                                          ;REPLACE ERROR CALL WITH
10199                                                          ;'BR 3$'' = 000740
10200  042756  062705  000002  8$:     ADD     #2,R5           ;POINT R5 TO THE NEXT PDR TO BE TESTED
10201  042762  062703  020000          ADD     #20000,R3       ;CHANGE VIRT. ADDR TO REF. NEXT PDR
10202  042766  077444                  SOB     R4,3$           ;LOOP BACK TO 3$ UNTIL ALL 8 PDR'S TESTED
10203  042770  004767  004764          JSR     PC,TON          ;TURN T-BIT BACK ON FOR NEXT TEST
10204  042774  005067  134776          CLR     PSW             ;BACK TO KERNEL MODE BEFORE LEAVING
10205
10206                          ;*****************************************************************************
10207                          ;TEST 400       TEST 'W-BIT'' SPECIAL CASES
10208                          ;*****************************************************************************
10209  043000                  TS400:
10210
10211  043000  004767  004720  1$:     JSR     PC,TOFF         ;TURN OFF T-BIT TRAPPING FOR THIS TEST
10212  043004  012701  077406          MOV     #77406,R1       ;PUT 'W-BIT OFF'' VALUE FOR PDR IN R1
10213  043010  010167  127302  2$:     MOV     R1,KIPDR7       ;LOAD KERNEL PDR 7 TO CLEAR W-BIT
10214  043014  016700  134552          MOV     SR0,R0          ;READ PRESENT CONTENTS OF STATUS REG. 0.
10215  043020  010067  134546          MOV     R0,SR0          ;WRITE PRESENT CONTENTS OF SR0 BACK TO ITSELF
10216  043024  016702  127266          MOV     KIPDR7,R2       ;READ CONTENTS OF KIPDR7 INTO R2
10217  043030  020102                  CMP     R1,R2           ;WAS W-BIT LEFT CLEARED?
10218  043032  001401                  BEQ     3$
10219  043034  104000                  EMT                     ;W-BIT IN KIPDR7 SET WHEN SR0 WAS WRITTEN TO
10220                                                          ;FOR TIGHTER SCOPE LOOP
10221                                                          ;REPLACE ERROR CALL WITH
10222                                                          ;'BR 2$'' = 000765
10223  043036  010167  127252  3$:     MOV     R1,KIPDR6       ;LOAD KERNEL PDR6 WITH 77406 TO CLEAR W-BIT
10224  043042  012767  043054  134734  MOV     #4$,ERRVEC      ;SET UP LOC. 4 TO 4$ FOR ODD ADDR. ABORT
10225  043050  005037  140000          CLR     @#140000        ;CAUSE TIMEOUT ABORT THRU LOC. 4
10226  043054  012706  001000  4$:     MOV     #KERSTK,KSP     ;RESTORE THE STACK POINTER
10227  043060  016702  127230          MOV     KIPDR6,R2       ;READ KIPDR6 INTO R2
10228  043064  052701  000100          BIS     #100,R1         ;R1-77506
10229  043070  020102                  CMP     R1,R2           ;WAS W-BIT SET?
10230  043072  001401                  BEQ     5$
```

```
10231  043074  104000                          EMT                     ;W-BIT WAS NOT SET DURING A TIMEOUT ABORT
10232                                                                   ;FOR TIGHTER SCOPE LOOP
10233                                                                   ;REPLACE ERROR CALL WITH
10234                                                                   ;'BR 3$'' = 000757
10235  043076  010167  127212                  5$:     MOV     R1,KIPDR6       ;RESTORE KIPDR6 TO 77406
10236  043102  012767  001400  127244                  MOV     #1400,KIPAR6    ;RESTORE KIPAR6 TO 1400
10237  043110  012767  021216  134666                  MOV     #T04,ERRVEC     ;RESTORE NORMAL CPU TRAP ROUTINE TO LOC.4
10238  043116  004767  004636                          JSR     PC,TON          ;TURN T-BIT TRAPPING BACK ON
10239
10240                                          ;*****************************************************************************
10241                                          ;*
10242                                          ;*      THE NEXT THREE (3) TESTS CAUSE MEMORY MANAGEMENT ERRORS
10243                                          ;*      TO CHECK THE ABILITY OF STATUS REGISTER 0 TO RECORD KT
10244                                          ;*      ERRORS AND THE ABILITY OF STATUS REGISTER 2 TO LOCK UP THE
10245                                          ;*      VIRTUAL ADDR. OF THE INSTRUCTION THAT CAUSED THE ERROR.
10246                                          ;*      THE BITS OF SR2 ARE CHECKED AND BITS <15:13>, <6:5>, AND <3:0>
10247                                          ;*      ARE CHECKED IN SR0.  SO THE SR0 AND SR2 LOGIC AND THE
10248                                          ;*      KT ERROR LOGIC ARE CHECKED.
10249                                          ;*
10250                                          ;*****************************************************************************
10251
10252                                          ;*****************************************************************************
10253                                          ;TEST 401       NON-RESIDENT ABORT TEST (ACF=0&4)
10254                                          ;*****************************************************************************
10255  043122                                  TS401:
10256
10257  043122  012700  001400                  1$:     MOV     #1400,R0        ;LOAD DATA FOR PAR'S INTO R0
10258  043126  010067  127214                          MOV     R0,KIPAR3       ;MAP KERNEL PAR'S 3&4 TO 24-28K
10259  043132  010067  127212                          MOV     R0,KIPAR4
10260  043136  010067  134504                          MOV     R0,UIPAR3       ;MAP USER PAR'S 3&4 TO 24-28K
10261  043142  010067  134502                          MOV     R0,UIPAR4
10262  043146  012767  077406  127132                  MOV     #77406,KIPDR3   ;MAP KERNEL PDR 3  128 BLKS, READ-WRITE
10263  043154  012767  077406  134424                  MOV     #77406,UIPDR3   ;MAP USER PDR 3  128 BLKS, READ-WRITE
10264  043162  012700  060000                          MOV     #60000,R0       ;LOAD VIRTUAL ADDR. TO REFERENCE PDR3 INTO R0
10265  043166  012701  100000                          MOV     #100000,R1      ;LOAD VIRTUAL ADDR. TO REFERENCE PDR4 INTO R1
10266  043172  012703  100011                          MOV     #100011,R3      ;LOAD R3 WITH WHAT SR0 SHOULD READ - N.R., KERNEL, PG.4
10267  043176  012702  077400                          MOV     #77400,R2       ;LOAD ACF=0 (NON-RESIDENT) PDR VALUE IN R2
10268  043202  012767  043236  135040          2$:     MOV     #5$,MMVEC       ;POINT MEM. MGMT. TRAP VECTOR TO 5$ BELOW
10269  043210  010267  127074                          MOV     R2,KIPDR4       ;LOAD ACF TEST VALUE INTO KIPDR4
10270  043214  010267  134370                          MOV     R2,UIPDR4       ;LOAD ACF TEST VALUE INTO UIPDR4
10271  043220  005010                  3$:     CLR     (R0)            ;CLEAR PHYS. LOC. 140000 USING PDR3
10272  043222  016767  134550  174010                  MOV     PSW,$TMP0       ;SAVE PSW IN CASE OF ERROR
10273  043230  005211                  4$:     INC     (R1)            ;TRY TO REF. IT USING PDR4 - SHOULD TRAP TO 5$
10274  043232  001462                          BEQ     TS402
10275  043234  104000                          EMT                     ;MEM. MGMT. ABORT DID NOT OCCUR
10276                                                                   ;FOR TIGHTER SCOPE LOOP
10277                                                                   ;REPLACE ERROR CALL WITH
10278                                                                   ;'BR 3$'' = 000772
10279  043236  062706  000004          5$:     ADD     #4,SP           ;RESTORE STACK POINTER
10280  043242  005710                          TST     (R0)            ;DID INSTRUCTION GET ABORTED & NOT EXECUTE
10281  043244  001401                          BEQ     6$
10282  043246  104000                          EMT                     ;INSTRUCTION WAS NOT ABORTED, LOC. GOT CHANGED
10283                                                                   ;FOR TIGHTER SCOPE LOOP
10284                                                                   ;REPLACE ERROR CALL WITH
10285                                                                   ;'BR 3$'' = 000764
10286  043250  016767  134316  173754  6$:     MOV     SR0,WASSR0      ;READ STATUS REGISTER 0
```

```
10287  043256  016767  134314  173750          MOV     SR2,WASSR2      ;READ STATUS REGISTER 2
10288  043264  020367  173742                  CMP     R3,WASSR0       ;DID SR0 REPORT NON-RESIDENT ERROR CORRECTLY?
10289  043270  001401                          BEQ     7$
10290  043272  104000                          EMT                     ;SR0 DID NOT REPORT NON-RES. ERROR CORRECTLY
10291                                                                   ;FOR TIGHTER SCOPE LOOP
10292                                                                   ;REPLACE ERROR CALL WITH
10293                                                                   ;'BR 3$'' = 000752
10294  043274  012704  043230          7$:     MOV     #4$,R4          ;LOAD R4 WITH WHAT SR2 SHOULD READ
10295  043300  020467  173730                  CMP     R4,WASSR2       ;DID SR2 LOCKUP RIGHT VIRTUAL ADDR. (=4$)?
10296  043304  001401                          BEQ     8$
10297  043306  104000                          EMT                     ;SR2 DID NOT LOCK VIRTUAL ADDR. OF NON-RES. ERROR
10298                                                                   ;FOR TIGHTER SCOPE LOOP
10299                                                                   ;REPLACE ERROR CALL WITH
10300                                                                   ;'BR 3$'' = 000744
10301  043310  042767  160000  134254  8$:     BIC     #160000,SR0     ;CLEAR THE ERROR BITS IN SR0
10302  043316  032767  140000  173714          BIT     #140000,$TMP0   ;HAS ACF=0&4 BEEN TESTED IN USER YET
10303  043324  001006                          BNE     9$              ;BRANCH IF YES
10304  043326  012703  100151                  MOV     #100151,R3      ;LOAD R3 WITH WHAT SR0 SHOULD READ - N.R., USER, PG.4
10305  043332  012767  140000  134436          MOV     #140000,PSW     ;GO TO USER MODE
10306  043340  000720                          BR      2$              ;REPEAT TEST IN USER MODE
10307  043342  022702  077404          9$:     CMP     #77404,R2       ;HAS ACF=4 BEEN TESTED YET?
10308  043346  001407                          BEQ     10$             ;BRANCH IF YES
10309  043350  012702  077404                  MOV     #77404,R2       ;THEN LOAD ACF=4 (NON-RES) PDR VALUE IN R2
10310  043354  012703  100011                  MOV     #100011,R3      ;LOAD R3 WITH WHAT SR0 SHOULD READ-N.R.,KERNEL,PG. 4
10311  043360  005067  134412                  CLR     PSW             ;GO BACK TO KERNEL MODE
10312  043364  000706                          BR      2$              ;GO BACK & TEST ACF=4 IN SAME MODE
10313  043370  005067  134404          10$:    CLR     PSW             ;GO BACK TO KERNEL MODE BEFORE LEAVING
10314  043372  012767  021236  134650          MOV     #T0250,MMVEC    ;RESTORE ADDRESS OF NORMAL MEMORY
10315                                                                   ;MANAGEMENT ERROR ROUTINE TO MMVEC
10316
10317                          ;*****************************************************************************************
10318                          ;TEST 402       READ-ONLY ABORT TEST (ACF=2)
10319                          ;*****************************************************************************************
10320  043400                  TS402:
10321  043400                  1$:                                     ;KERNEL & USER PAR'S 3 & 4 AND PDR 3
10322                                                                  ;ARE SETUP FROM LAST TEST
10323  043400  012700  060000          MOV     #60000,R0               ;LOAD VIRTUAL ADDR. TO REFERENCE PDR3 INTO R0
10324  043404  012701  100000          MOV     #100000,R1              ;LOAD VIRTUAL ADDR. TO REFERENCE PDR4 INTO R1
10325  043410  012703  020011          MOV     #20011,R3               ;LOAD R3 WITH WHAT SR0 SHOULD READ - R/O, KERNEL, PG.4
10326  043414  012702  077402          MOV     #77402,R2               ;LOAD ACF=2 (READ-ONLY) PDR VALUE IN R2
10327  043420  012767  043452  134622  2$:     MOV     #5$,MMVEC       ;POINT MEM. MGMT. TRAP VECTOR TO 5$ BELOW
10328  043426  010267  126656          MOV     R2,KIPDR4               ;LOAD ACF=2 INTO KIPDR4
10329  043432  010267  134152          MOV     R2,UIPDR4               ;LOAD ACF=2 INTO UIPDR4
10330  043436  005010                  3$:     CLR     (R0)            ;CLEAR PHYS. LOC. 140000 USING PDR3
10331  043440  016767  134332  173572          MOV     PSW,$TMP0       ;SAVE PSW IN CASE OF ERROR
10332  043446  005211                  4$:     INC     (R1)            ;TRY TO WRITE USING PDR4 - SHOULD TRAP TO 5$
10333  043450  104000                          EMT                     ;MEM. MGMT. ABORT DID NOT OCCUR
10334                                                                  ;FOR TIGHTER SCOPE LOOP
10335                                                                  ;REPLACE ERROR CALL WITH
10336                                                                  ;'BR 3$'' = 000772
10337  043452  062706  000004          5$:     ADD     #4,SP           ;RESTORE STACK POINTER
10338  043456  005710                          TST     (R0)            ;DID INSTRUCTION GET ABORTED & NOT EXECUTE
10339  043460  001401                          BEQ     6$
10340  043462  104000                          EMT                     ;INSTRUCTION WAS NOT ABORTED, LOC. GOT CHANGED
10341                                                                  ;FOR TIGHTER SCOPE LOOP
10342                                                                  ;REPLACE ERROR CALL WITH
```

```
10343
10344  043464  016767  134102  173540  6$:   MOV   SR0,WASSR0     ;'BR 3$'' = 000764
10345  043472  016767  134100  173534        MOV   SR2,WASSR2     ;READ STATUS REG. 0
10346  043500  020367  173526              CMP   R3,WASSR0      ;READ STATUS REG. 2
10347  043504  001401                      BEQ   7$             ;DID SR0 REPORT READ-ONLY ERROR CORRECTLY?
10348  043506  104000                      EMT                  ;SR0 DID NOT REPORT R/O ERROR CORRECTLY
10349                                                           ;FOR TIGHTER SCOPE LOOP
10350                                                           ;REPLACE ERROR CALL WITH
10351                                                           ;'BR 3$'' = 000752
10352  043510  012704  043446        7$:   MOV   #4$,R4         ;LOAD R4 WITH WHAT SR2 SHOULD READ
10353  043514  020467  173514              CMP   R4,WASSR2      ;DID SR2 LOCKUP RIGHT VIRTUAL ADDR. (=4$)?
10354  043520  001401                      BEQ   8$
10355  043522  104000                      EMT                  ;SR2 DID NOT LOCKUP VIRTUAL ADDR. OF R/O ERROR
10356                                                           ;FOR TIGHTER SCOPE LOOP
10357                                                           ;REPLACE ERROR CALL WITH
10358                                                           ;'BR 3$'' = 000744
10359  043524  042767  160000  134040  8$:   BIC   #160000,SR0    ;CLEAR THE ERROR BITS IN SR0
10360  043532  032767  140000  173500        BIT   #140000,$TMP0  ;HAS ACF=2 BEEN TESTED IN USER MODE?
10361  043540  001006                      BNE   9$             ;BRANCH IF YES
10362  043542  012703  020151              MOV   #20151,R3      ;LOAD R3 WITH WHAT SR0 SHOULD READ-R/O, USER, PG.4
10363  043546  012767  140000  134222      MOV   #140000,PSW    ;GO TO USER MODE
10364  043554  000721                      BR    2$             ;REPEAT TEST IN USER MODE
10365  043556  005067  134214        9$:   CLR   PSW            ;GO BACK TO KERNEL MODE BEFORE LEAVING
10366  043562  012767  021236  134460      MOV   #T0250,MMVEC   ;RESTORE ADDRESS OF NORMAL MEMORY
10367                                                           ;MANAGEMENT ERROR ROUTINE TO MMVEC.
10368
10369
10370
10371                          ;NOTE:  MACRO MSG31A WAS DELETED AS IT DIDN'T APPLY TO F11.
10372
10373
10374
10375
10376
10377
10378
10379
10380                          ;*****************************************************************************
10381                          ;*
10382                          ;*      THE NEXT TWO (2) TESTS WILL BE CHECKING THE PAGE LENGTH
10383                          ;*      COMPARATORS AND SOME MORE OF THE KT ERROR DETECTION
10384                          ;*      AND STATUS LOGIC.  THE PAGE LENGTH FIELD (PLF) IN KERNEL
10385                          ;*      PDR 4 IS VARIED AND FOR EVERY PLF, THREE (3) VIRTUAL
10386                          ;*      ADDRESSES ARE READ.  WHILE USING BOTH UPWARD & DOWNWARD PAGE
10387                          ;*      EXPANSION, ONE OF THOSE THREE VIRTUAL ADDRESSES WILL CAUSE A
10388                          ;*      ''PAGE LENGTH ABORT'' WHILE THE OTHER TWO WON'T.
10389                          ;*
10390                          ;*      STATUS REGISTER 0 & 2 ARE CHECKED WHEN THE PAGE LENGTH
10391                          ;*      ABORT DOES OCCUR TO SEE THAT THE ABORT IS REPORTED AND THAT
10392                          ;*      THE VIRTUAL ADDRESS OF THE INSTRUCTION THAT CAUSED THE ABORT
10393                          ;*      IS LOCKED UP.          ;
10394                          ;*
10395                          ;*****************************************************************************
10396
10397
10398
```

```
10399
10400
10401
10402   043570
10403   043570  012767  077406  126510  1$:     MOV     #77406,KIPDR3   ;MAKE SURE PDR3 IS DESCRIBED AS R/W
10404   043576  012767  077406  126506          MOV     #77406,KIPDR5   ;MAKE SURE PDR5 IS DESCRIBED AS R/W
10405   043604  012700  044004                  MOV     #DALTB1,R0      ;DAL TABLE FOR VIRTUAL ADDR'S. TO SELECT PDR4.
10406   043610  012704  044022                  MOV     #PDRTB1,R4      ;PDR TABLE FOR PDR4 (COINCIDES WITH DAL TABLE).
10407   043614  012701  000006                  MOV     #6,R1           ;SET UP LOOP COUNTER.
10408   043620  012767  043762  134422          MOV     #9$,MMVEC       ;SETUP M.M. TRAP VECTOR FOR UNEXPECTED ABORTS
10409   043626  012706  001000                  MOV     #KERSTK,KSP     ;MAKE SURE STACK POINTER IS ALL SET UP
10410
10411                                           ;TEST NON-ABORT CASES (VBA < OR = PLF)
10412   043632  012467  126452          2$:     MOV     (R4)+,KIPDR4    ;LOAD KIPDR4 WITH PAGE LENGTH VALUE
10413   043636  005730                          TST     @(R0)+          ;ACCESS VIRTUAL ADDR. (VBA < OR = PLF)
10414                                                                   ;NO ABORT SHOULD OCCUR!!!
10415   043640  077104                          SOB     R1,2$           ;DONE?...NO- TEST NEXT COMBINATION OF DAL & PDR.
10416
10417                                           ;TEST ABORT CASES (VBA > PLF)
10418   043642  012701  000005          3$:     MOV     #5,R1           ;SET UP LOOP COUNTER.
10419   043646  012700  044040                  MOV     #DALTB2,R0      ;DAL TABLE
10420   043652  012704  044054                  MOV     #PDRTB2,R4      ;PDR TABLE
10421   043656  012767  043676  134364          MOV     #6$,MMVEC       ;SETUP M.M. TRAP VECTOR FOR EXPECTED ABORT
10422
10423   043664  012467  126420          4$:     MOV     (R4)+,KIPDR4    ;LOAD KIPDR4 WITH PAGE LENGHT VALUE
10424   043670  005730                  5$:     TST     @(R0)+          ;ACCESS VIRTUAL ADDR. (VBA > PLF - ABORT TO 6$)
10425   043672  001476                          BEQ     TS404
10426   043674  104000                          EMT                     ;EXPECTED PAGE LENGTH ABORT DID NOT OCCUR
10427                                                                   ;FOR TIGHTER SCOPE LOOP
10428                                                                   ;REPLACE ERROR CALL WITH
10429                                                                   ;'BR 5$'' = 000776
10430   043676  012706  001000          6$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER FOLLOWING ABORT
10431   043702  016767  133664  173322          MOV     SR0,WASSR0      ;READ M.M. STATUS REG. 0
10432   043710  016767  133662  173316          MOV     SR2,WASSR2      ;READ M.M. STATUS REG. 2
10433   043716  012702  040011                  MOV     #40011,R2       ;PUT EXPECTED SR0 CONTENTS IN R2
10434   043722  020267  173304                  CMP     R2,WASSR0       ;DID SR0 REPORT PG. LENGTH ABORT, PAGE 4, KERNEL?
10435   043726  001401                          BEQ     7$
10436   043730  104000                          EMT                     ;SR0 DID NOT REPORT PG. LENGTH ABORT CORRECTLY
10437                                                                   ;FOR TIGHTER SCOPE LOOP
10438                                                                   ;REPLACE ERROR CALL WITH
10439                                                                   ;'BR 5$'' = 000757
10440   043732  012703  043670          7$:     MOV     #5$,R3          ;PUT EXPECTED SR2 CONTENTS IN R3
10441   043736  020367  173272                  CMP     R3,WASSR2       ;DID SR2 LOCKUP VIRT. ADDR. OF ABORTED INSTRUCTION?
10442   043742  001401                          BEQ     8$
10443   043744  104000                          EMT                     ;SR2 DID NOT LOCKUP VIRT. ADDR. OF ABORT CORRECTLY
10444                                                                   ;FOR TIGHTER SCOPE LOOP
10445                                                                   ;REPLACE ERROR CALL WITH
10446                                                                   ;'BR 5$'' = 000751
10447   043746  042767  160000  133616  8$:     BIC     #160000,SR0     ;CLEAR ERROR BITS IN SR0
10448   043754  077135                          SOB     R1,4$           ;DONE?..NO - GET NEXT DAL & PDR PAIR
10449   043756  000167  000010                  JMP     10$             ;YES...
10450   043762  042767  160000  133602  9$:     BIC     #160000,SR0     ;CLEAR ERROR BITS IN SR0
10451   043770  104000                          EMT                     ;GOT PG. LENGTH ABORT BEFORE IT WAS EXPECTED
10452                                                                   ;FOR TIGHTER SCOPE LOOP
10453                                                                   ;REPLACE ERROR CALL WITH
10454                                                                   ;A 'NOP'' = 240
```

```
10455
10456  043772  012767  021236  134250  10$:    MOV     #T0250,MMVEC    ;RESTORE NORMAL M.M. TRAP HANDLER
10457                                                                    ;ADDRESS TO M.M. TRAP VECTOR
10458  044000  000167  000064            JMP     TS404           ;GET TO NEXT TEST
10459
10460                                  ;DAL TABLE FOR UPWARD EXPANSION (NON-ABORT CASES)
10461  044004  100000                  DALTB1: 100000
10462  044006  106100                          106100
10463  044010  102300                          102300
10464  044012  102500                          102500
10465  044014  113700                          113700
10466  044016  104600                          104600
10467  044020  117700                          117700
10468
10469                                  ;PDR TABLE FOR KPDR4 (NON-ABORT CASES)
10470  044022  000006                  PDRTB1: 000006
10471  044024  052006                          052006
10472  044026  045006                          045006
10473  044030  052006                          052006
10474  044032  074406                          074406
10475  044034  025006                          025006
10476  044036  077406                          077406
10477
10478                                  ;DAL TABLE (ABORT CASES)
10479  044040  100100                  DALTB2: 100100
10480  044042  110100                          110100
10481  044044  116600                          116600
10482  044046  112700                          112700
10483  044050  117000                          117000
10484  044052  117700                          117700
10485
10486                                  ;PDR TABLE (ABORT CASES)
10487  044054  000006                  PDRTB2: 000006
10488  044056  030406                          030406
10489  044060  046406                          046406
10490  044062  042006                          042006
10491  044064  073406                          073406
10492  044066  077006                          077006
10493
10494
10495                                  ;*****************************************************************************************
10496                                  ;TEST 404       PAGE LENGTH FAULTS-DOWNWARD EXPANSION
10497                                  ;*****************************************************************************************
10498  044070                          TS404:
10499  044070  012700  044270          1$:     MOV     #DALTB3,R0      ;DAL TABLE FOR VIRTUAL ADDR'S. TO SELECT PDR4.
10500  044074  012704  044306                  MOV     #PDRTB3,R4      ;PDR TABLE FOR PDR4 (COINCIDES WITH DAL TABLE).
10501  044100  012701  000006                  MOV     #6,R1           ;SET UP LOOP COUNTER.
10502  044104  012767  044246  134136          MOV     #9$,MMVEC       ;SETUP M.M. TRAP VECTOR FOR UNEXPECTED ABORTS
10503  044112  012706  001000                  MOV     #KERSTK,KSP     ;MAKE SURE STACK POINTER IS ALL SET UP
10504
10505                                  ;TEST NON-ABORT CASES (VBA > OR = PLF)
10506  044116  012467  126166          2$:     MOV     (R4)+,KIPDR4    ;LOAD KIPDR4 WITH PAGE LENGTH VALUE
10507  044122  005730                          TST     @(R0)+          ;ACCESS VIRTUAL ADDR. (VBA > OR = PLF)
10508                                                                  ;NO ABORT SHOULD OCCUR!!!
10509  044124  077104                          SOB     R1,2$           ;DONE?...NO- TEST NEXT COMBINATION OF DAL & PDR.
10510
```

```
10511                                          ;TEST ABORT CASES (VBA < PLF)
10512  044126  012701  000005          3$:     MOV     #5,R1          ;SET UP LOOP COUNTER.
10513  044132  012700  044324                  MOV     #DALTB4,R0     ;DAL TABLE
10514  044136  012704  044340                  MOV     #PDRTB4,R4     ;PDR TABLE
10515  044142  012767  044162  134100          MOV     #6$,MMVEC      ;SETUP M.M. TRAP VECTOR FOR EXPECTED ABORT
10516
10517  044150  012467  126134          4$:     MOV     (R4)+,KIPDR4   ;LOAD KIPDR4 WITH PAGE LENGHT VALUE
10518  044154  005730                  5$:     TST     @(R0)+         ;ACCESS VIRTUAL ADDR. (VBA < PLF - ABORT TO 6$)
10519  044156  001476                          BEQ     TS405
10520  044160  104000                          EMT                    ;EXPECTED PAGE LENGTH ABORT DID NOT OCCUR
10521                                                                 ;FOR TIGHTER SCOPE LOOP
10522                                                                 ;REPLACE ERROR CALL WITH
10523                                                                 ;'BR 5$' = 000776
10524  044162  012706  001000          6$:     MOV     #KERSTK,KSP    ;RESTORE STACK POINTER FOLLOWING ABORT
10525  044166  016767  133400  173036          MOV     SR0,WASSR0     ;READ M.M. STATUS REG. 0
10526  044174  016767  133376  173032          MOV     SR2,WASSR2     ;READ M.M. STATUS REG. 2
10527  044202  012702  040011                  MOV     #40011,R2      ;PUT EXPECTED SR0 CONTENTS IN R2
10528  044206  020267  173020                  CMP     R2,WASSR0      ;DID SR0 REPORT PG. LENGTH ABORT, PAGE 4, KERNEL?
10529  044212  001401                          BEQ     7$
10530  044214  104000                          EMT                    ;SR0 DID NOT REPORT PG. LENGTH ABORT CORRECTLY
10531                                                                 ;FOR TIGHTER SCOPE LOOP
10532                                                                 ;REPLACE ERROR CALL WITH
10533                                                                 ;'BR 5$' = 000757
10534  044216  012703  044154          7$:     MOV     #5$,R3         ;PUT EXPECTED SR2 CONTENTS IN R3
10535  044222  020367  173006                  CMP     R3,WASSR2      ;DID SR2 LOCKUP VIRT. ADDR. OF ABORTED INSTRUCTION?
10536  044226  001401                          BEQ     8$
10537  044230  104000                          EMT                    ;SR2 DID NOT LOCKUP VIRT. ADDR. OF ABORT CORRECTLY
10538                                                                 ;FOR TIGHTER SCOPE LOOP
10539                                                                 ;REPLACE ERROR CALL WITH
10540                                                                 ;'BR 5$' = 000751
10541  044232  042767  160000  133332  8$:     BIC     #160000,SR0    ;CLEAR ERROR BITS IN SR0
10542  044240  077135                          SOB     R1,4$          ;DONE?..NO - GET NEXT DAL & PDR PAIR
10543  044242  000167  000010                  JMP     10$            ;YES...
10544  044246  042767  160000  133316  9$:     BIC     #160000,SR0    ;CLEAR ERROR BITS IN SR0
10545  044254  104000                          EMT                    ;GOT PG. LENGTH ABORT BEFORE IT WAS EXPECTED
10546                                                                 ;FOR TIGHTER SCOPE LOOP
10547                                                                 ;REPLACE ERROR CALL WITH
10548                                                                 ;A 'NOP' = 000240
10549
10550  044256  012767  021236  133764  10$:     MOV     #T0250,MMVEC   ;RESTORE NORMAL M.M. TRAP HANDLER
10551                                                                 ;ADDRESS TO M.M. TRAP VECTOR
10552  044264  000167  000064                  JMP     TS405          ;GET TO NEXT TEST
10553
10554                                          ;DAL TABLE FOR DOWNWARD EXPANSION (NON-ABORT CASES)
10555  044270  117700                  DALTB3: 117700
10556  044272  111600                          111600
10557  044274  115400                          115400
10558  044276  115200                          115200
10559  044300  104000                          104000
10560  044302  113100                          113100
10561  044304  100000                          100000
10562
10563                                          ;PDR TABLE (NON-ABORT CASES)
10564  044306  077416                  PDRTB3: 77416
10565  044310  025416                          25416
10566  044312  032416                          32416
```

```
10567  044314  025416                      25416
10568  044316  003016                      03016
10569  044320  052416                      52416
10570  044322  000016                      00016
10571
10572                                      ;DAL TABLE (ABORT CASES)
10573  044324  117600                      DALTB4: 117600
10574  044326  107600                              107600
10575  044330  101100                              101100
10576  044332  105000                              105000
10577  044334  100700                              100700
10578  044336  100000                              100000
10579
10580                                      ;PDR TABLE (ABORT CASES)
10581  044340  077416                      PDRTB4: 77416
10582  044342  047016                              47016
10583  044344  031016                              31016
10584  044346  035416                              35416
10585  044350  004016                              04016
10586  044352  000416                              00416
10587
10588
10589
10590                                      ;*****************************************************************************
10591                                      ;TEST 405       SR2 BIT TEST
10592                                      ;*****************************************************************************
10593  044354                              TS405:
10594  044354  012767  001400  125764      1$:     MOV     #1400,KIPAR3    ;BE SURE PAR3 IS MAPPED TO 24-28K
10595  044362  012767  001400  125760              MOV     #1400,KIPAR4    ;BE SURE PAR4 IS MAPPED TO 24-28K
10596  044370  012767  077406  125710              MOV     #77406,KIPDR3   ;MAP PAGE 3  128 BLOCKS, R/W
10597  044376  012767  077402  125704              MOV     #77402,KIPDR4   ;MAP PAGE 4  128 BLOCKS, READ-ONLY
10598  044404  012700  060002                      MOV     #60002,R0       ;LOAD R0 WITH VIRTUAL ADDR. WHICH USE'S PDR3
10599  044410  012701  100002                      MOV     #100002,R1      ;LOAD R1 WITH VIRTUAL ADDR. WHICH USES PDR4
10600  044414  012767  044442  133626              MOV     #3$,MMVEC       ;SET M.M. TRAP VECTOR TO 3$
10601  044422  012720  010727              2$:     MOV     #010727,(R0)+   ;LOAD 'MOV PC,(PC)+'' INSTRUCTION AT ADDR.
10602  044426  005020                              CLR     (R0)+           ;       REACHED THRU PDR/PAR 4.
10603  044430  012720  000137                      MOV     #000137,(R0)+   ;LOAD ''JMP @#3$'' INSTRUCTION AT VIRT. ADDR.
10604  044434  012710  044442                      MOV     #3$,(R0)        ;       IN CASE R/O VIOL. DOES NOT ABORT
10605  044440  010107                              MOV     R1,PC           ;TRANSFER PROGRAM EXECUTION TO ''PAGE 4 INSTRUCTIONS''
10606  044442  012706  001000              3$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER
10607  044446  016767  133124  172560              MOV     SR2,WASSR2      ;READ CONTENTS OF STATUS REG 2
10608  044454  020167  172554                      CMP     R1,WASSR2       ;WAS ADDR. OF ''RELOCATED - R/O ABORT'' LOCKED UP?
10609  044460  001401                              BEQ     4$
10610  044462  104000                              EMT                     ;SR2 DID NOT LOCK UP VIRTUAL ADDR. OF R/O VIOL.
10611                                              ;FOR TIGHTER SCOPE LOOP
10612                                              ;REPLACE ERROR CALL WITH
10613                                              ;''BR 2$'' = 000757
10614  044464  042767  160000  133100      4$:     BIC     #160000,SR0     ;CLEAR THE ERROR BITS IN SR0
10615  044472  060101                              ADD     R1,R1           ;SETUP TO FORM NEXT VIRTUAL ADDRESS
10616  044474  010100                              MOV     R1,R0           ;SETUP R0 TO FORM NEXT VIRT. ADDR. TO LOAD
10617  044476  052701  100000                      BIS     #100000,R1      ;FORM VIRTUAL ADDR. THAT SHOULD BE LOCKED UP NEXT
10618  044502  052700  060000                      BIS     #60000,R0       ;POINT R0 TO NEXT VIRT. ADDR. TO LOAD
10619  044506  020127  110000                      CMP     R1,#110000      ;HAVE ALL VBA'S 100000-110000 BEEN TESTED?
10620  044512  101743                              BLOS    2$              ;BRANCH IF NO
10621
10622  044514  012767  077406  125566              MOV     #77406,KIPDR4   ;RESTORE PDR4 TO R/W ACCESS
```

```
10623  044522  012767  021236  133520          MOV     #T0250,MMVEC    ;RESTORE ADDRESS OF NORMAL M.M.
10624                                                                   ;TRAP HANDLER TO M.M. VECTOR
10625
10626
10627                                  ;*****************************************************************************
10628                                  ;TEST 406       MORE CHECKS OF SR0 & SR2
10629                                  ;*****************************************************************************
10630  044530                          TS406:
10631  044530  012767  001400  125614  1$:     MOV     #1400,KIPAR5    ;MAP KERNEL PAGE 5 TO 24-28K
10632  044536  012767  000406  125544          MOV     #406,KIPDR4     ;SETUP PDR4 FOR PAGE LENGTH ABORT
10633  044544  012767  077402  125540          MOV     #77402,KIPDR5   ;SETUP PDR5 FOR R/O ABORT
10634  044552  016767  133020  172454  2$:     MOV     SR2,WASSR2      ;READ SR2 TO SEE IF ITS TRACKING
10635  044560  012701  044552                  MOV     #2$,R1          ;PUT EXPECTED VIRTUAL PC IN R1
10636  044564  020167  172444                  CMP     R1,WASSR2       ;DID SR2 CONTAIN VIRTUAL PC AT 2$?
10637  044570  001401                          BEQ     4$
10638  044572  104000                          EMT                     ;SR2 NOT TRACKING CORRECTLY
10639                                                                  ;FOR TIGHTER SCOPE LOOP
10640                                                                  ;REPLACE ERROR CALL WITH
10641                                                                  ;'BR 2$'' = 000767
10642  044574  016767  132776  172432  4$:     MOV     SR2,WASSR2      ;READ SR2 TO SEE IF ITS TRACKING
10643  044602  012701  044574                  MOV     #4$,R1          ;PUT EXPECTED VIRTUAL PC IN R1
10644  044606  020167  172422                  CMP     R1,WASSR2       ;DID SR2 CONTAIN VIRTUAL PC AT 4$
10645  044612  001401                          BEQ     6$
10646  044614  104000                          EMT                     ;SR2 NOT TRACKING CORRECTLY
10647                                                                  ;FOR TIGHTER SCOPE LOOP
10648                                                                  ;REPLACE ERROR CALL WITH
10649                                                                  ;'BR 4$'' = 000767
10650  044616  012767  044634  133424  6$:     MOV     #7$,MMVEC       ;PUT ADDRESS OF 7$ IN M.M. TRAP VECTOR
10651  044624  005067  172412                  CLR     $TMP1           ;CLEAR ERROR INDICATOR
10652  044630  005237  100500                  INC     @#100500        ;CAUSE PAGE LENGTH ABORT - TRAP TO 7$
10653  044634  012706  001000          7$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER AFTER ABORT
10654  044640  016767  132726  172372          MOV     SR0,$TMP0       ;SAVE SR0'S INFORMATION ON PG. LGTH. ABORT
10655  044646  016767  132724  172370          MOV     SR2,$TMP2       ;SAVE SR2'S INFORMATION ON PG. LGTH. ABORT
10656  044654  012767  044666  133366          MOV     #8$,MMVEC       ;PUT ADDRESS OF 8$ IN M.M. TRAP VECTOR
10657  044662  005237  120000                  INC     @#120000        ;CAUSE R/O ABORT - TRAP TO 8$
10658  044666  012706  001000          8$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER AFTER ABORT
10659  044672  016767  132674  172332          MOV     SR0,WASSR0      ;READ SR0 FOLLWOING SECOND KT ABORT
10660  044700  016767  132672  172326          MOV     SR2,WASSR2      ;READ SR2 FOLLOWING SECOND KT ABORT
10661  044706  026767  172326  172316          CMP     $TMP0,WASSR0    ;IS SR0 STILL HOLDING INFO ON FIRST ABORT?
10662  044714  001402                          BEQ     9$              ;BRANCH IF YES
10663  044716  005267  172320                  INC     $TMP1           ;SET ERROR INDICATOR
10664  044722  026767  172316  172304  9$:     CMP     $TMP2,WASSR2    ;DOES SR2 STILL HOLD PC OF FIRST ABORT?
10665  044730  001402                          BEQ     10$             ;BRANCH IF YES
10666  044732  005267  172304                  INC     $TMP1           ;SET ERROR INDICATOR
10667  044736  005767  172300          10$:    TST     $TMP1           ;WERE SR0 OR SR2 CHANGED BY A SECOND ABORT?
10668  044742  001401                          BEQ     11$
10669  044744  104000                          EMT                     ;ONE OF STATUS REGS. CHANGED BY SECOND ABORT
10670                                                                  ;FOR TIGHTER SCOPE LOOP
10671                                                                  ;REPLACE ERROR CALL WITH
10672                                                                  ;'BR 6$'' = 000726
10673  044746  005067  172270          11$:    CLR     $TMP1           ;CLEAR ERROR INDICATOR
10674  044752  000005                          RESET                   ;EXECUTE A RESET, APPLYING AN ''INIT''
10675  044754  005067  132612                  CLR     SR0
10676  044760  016767  132606  172244          MOV     SR0,WASSR0      ;READ SR0
10677  044766  005767  172240                  TST     WASSR0          ;WAS SR0 CLEARED BY THE RESET?
10678  044772  001402                          BEQ     12$             ;BRANCH IF YES
```

```
10679  044774  005267  172242                  INC     $TMP1           ;SR0 NOT CLEARED BY A RESET
10680  045000  016767  132572  172226  12$:    MOV     SR2,WASSR2      ;READ SR2
10681  045006  022767  045000  172220          CMP     #12$,WASSR2     ;WAS SR2 UNLOCKED BY A RESET?
10682  045014  001402                          BEQ     13$             ;BRANCH IF YES
10683  045016  005267  172220                  INC     $TMP1           ;SR2 NOT UNLOCKED BY A RESET
10684  045022  005767  172214          13$:    TST     $TMP1           ;WERE SR0 & SR2 BOTH "RESET" BY A RESET?
10685  045026  001401                          BEQ     14$
10686  045030  104000                          EMT                     ;SR0 OR SR2 NOT "RESET" BY A RESET
10687                                                                  ;FOR TIGHTER SCOPE LOOP
10688                                                                  ;REPLACE ERROR CALL WITH
10689                                                                  ;"BR 6$" = 000676
10690  045032  012767  000001  132532  14$:    MOV     #1,SR0          ;TURN MEMORY MANAGEMENT BACK ON
10691  045040  016767  132532  172166  15$:    MOV     SR2,WASSR2      ;READ SR2 TO SEE IF ITS TRACKING AGAIN
10692  045046  012701  045040                  MOV     #15$,R1         ;PUT EXPECTED VIRTUAL PC IN R1
10693  045052  020167  172156                  CMP     R1,WASSR2       ;DID SR2 CONTAIN VIRTUAL PC AT 15$
10694  045056  001401                          BEQ     16$
10695  045060  104000                          EMT                     ;SR2 NOT TRACKING CORRECTLY
10696                                                                  ;FOR TIGHTER SCOPE LOOP
10697                                                                  ;REPLACE ERROR CALL WITH
10698                                                                  ;"BR 6$" = 000663
10699  045062  012767  077406  125220  16$:    MOV     #77406,KIPDR4   ;RESET PDR4 TO 128 BLKS, R/W
10700  045070  012767  077406  125214          MOV     #77406,KIPDR5   ;RESET PDR5 TO 128 BLKS, R/W
10701  045076  012767  021236  133144          MOV     #T0250,MMVEC    ;RESTORE ADDRESS OF NORMAL MEMORY
10702                                                                  ;MANAGEMENT TRAP ROUTINE TO M.M. VECTOR
10703
10704
10705                          ;*****************************************************************************
10706                          ;TEST 407       USER ABORT PICKS UP KERNEL SPACE VECTOR
10707                          ;*****************************************************************************
10708  045104                  TS407:
10709  045104  004767  002614          1$:     JSR     PC,TOFF         ;TURN OFF T-BIT TRAPPING FOR THIS TEST
10710  045110  005067  132662          2$:     CLR     PSW             ;GO TO KERNEL MODE
10711  045114  012706  001000                  MOV     #KERSTK,KSP     ;SETUP KERNEL STACK PTR.
10712  045120  012767  001400  132512          MOV     #1400,UIPAR0    ;MAP USER PAGE 0 TO 24K
10713  045126  012737  045176  000004          MOV     #4$,@#4         ;LOAD KERNEL VECTOR 4 (LOC.4) WITH 4$
10714  045134  012737  000340  000006          MOV     #340,@#6        ;LOAD VECTOR+2 WITH NEW PSW
10715  045142  012767  140000  132626          MOV     #140000,PSW     ;GO TO USER MODE
10716  045150  012706  000600                  MOV     #USESTK,USP     ;SETUP USER STACK PTR.
10717  045154  012737  045174  000004          MOV     #3$,@#4         ;LOAD USER VECTOR 4 (LOC. 60004) WITH 3$
10718  045162  012737  000340  000006          MOV     #340,@#6        ;LOAD VECTOR+2 WITH NEW PSW
10719  045170  005767  112604                  TST     160000          ;CAUSE TIMEOUT ERROR TRAP TO "4"
10720                                                                  ;SHOULD PICK UP NEW PC=4$ FROM KERNEL
10721                                                                  ;LOC. 4, NOT PC=3$ FROM USER LOC. 4 (=60004)
10722  045174                  3$:
10723  045174  104000                          EMT                     ;DID NOT TRAP THRU KERNEL SPACE
10724                                                                  ;FOR TIGHTER SCOPE LOOP
10725                                                                  ;REPLACE ERROR CALL WITH
10726                                                                  ;"BR 2$" = 000740
10727  045176  005067  132574          4$:     CLR     PSW             ;BE SURE BACK IN KERNEL MODE
10728  045202  012706  001000                  MOV     #KERSTK,KSP     ;RESTORE KERNEL S.P. IN CASE IT CHANGED
10729  045206  005067  132426                  CLR     UIPAR0          ;REMAP USER PAGE 0 TO 0-4K
10730  045212  012767  140000  132556          MOV     #140000,PSW     ;GO TO USER MODE
10731  045220  012706  000600                  MOV     #USESTK,USP     ;RESTORE USER STACK POINTER
10732  045224  005067  132546                  CLR     PSW             ;GO BACK TO KERNEL MODE
10733  045230  012737  021216  000004          MOV     #T04,@#4        ;RESTORE ADDR. OF NORMAL CPU TRAP HANDLER TO 4
10734  045236  004767  002516                  JSR     PC,TON          ;TURN T-BIT TRAPPING BACK ON
```

```
10735
10736
10737
10738
10739  045242
10740
10741  045242  012702  170000
10742  045246  010267  132524
10743  045252  012746  000340
10744  045256  012746  045264
10745  045262  000002
10746  045264  016701  132506
10747  045270  042701  007437
10748  045274  005067  132476
10749  045300  020201
10750  045302  001401
10751  045304  104000
10752
10753
10754
10755
10756
10757
10758
10759
10760  045306
10761  045306  012705  077006
10762  045312  010567  125000
10763  045316  012737  045336  000004
10764  045324  012737  045340  000250
10765  045332  005237  177700
10766  045336
10767  045336  104000
10768
10769
10770
10771  045340  012706  001000
10772  045344  016767  132222  171660
10773  045352  016767  132220  171654
10774  045360  012700  040017
10775  045364  020067  171642
10776  045370  001401
10777  045372  104000
10778
10779
10780
10781  045374  012701  045332
10782  045400  020167  171630
10783  045404  001401
10784  045406  104000
10785
10786
10787
10788  045410  042767  160000  132154
10789  045416  012737  021216  000004
10790  045424  012737  021236  000250

;*****************************************************************************
;TEST 410        RTI IN USER MODE DOES NOT CHANGE PSW
;*****************************************************************************
TS410:

        MOV     #170000,R2      ;LOAD 'PRESENT & EXPECTED'' PSW VALUE INTO R2
2$:     MOV     R2,PSW          ;GO TO USER MODE-PRIORITY 0
        MOV     #340,-(SP)      ;PUT A NEW PSW (PRIORITY=7) ON STACK
        MOV     #3$,-(SP)       ;PUT NEW PC ON THE STACK
        RTI                     ;DO AN RTI FROM USER MODE
3$:     MOV     PSW,R1          ;READ NEW PSW INTO R1
        BIC     #7437,R1        ;MASK OFF COND. CODE, T-BIT, AND UNUSED BITS
        CLR     PSW             ;GO BACK TO KERNEL MODE
        CMP     R2,R1           ;DID PSW STAY IN USER, PRIORITY=0?
        BEQ     TS411
        EMT                     ;PSW CHANGED BY AN RTI FROM USER
                                ;FOR A TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;'BR=2$'' = 000760

;*****************************************************************************
;TEST 411        KT ERROR SERVICED BEFORE TIMEOUT ERROR
;*****************************************************************************
TS411:
1$:     MOV     #77006,R5       ;LOAD PDR7 DATA INTO R5
        MOV     R5,KIPDR7       ;MAP PAGE 7 R/W  PLF=176
        MOV     #3$,@#4         ;SET CPU TRAP VECTOR TO ADDRESS OF 3$
        MOV     #4$,@#250       ;SET M.M. TRAP VECTOR TO ADDRESS OF 4$
2$:     INC     @#177700        ;CAUSE PLF ABORT AND POTENTIAL TIMEOUT
3$:
        EMT                     ;TRAPPED THRU CPU TRAP VECTOR BUT SHOULDN'T HAVE
                                ;FOR TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;'BR 2$'' = 000776
4$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER AFTER TRAPPING
        MOV     SR0,WASSR0      ;READ STATUS REG.0
5$:     MOV     SR2,WASSR2      ;READ STATUS REG. 2
        MOV     #40017,R0       ;LOAD EXPECTED SR0 CONTENTS INTO R0
        CMP     R0,WASSR0       ;SR0 PLF ERROR BIT SET?
        BEQ     6$
        EMT                     ;SR0 DIDN'T REPORT PLF ERROR
                                ;FOR TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;'BR 2$'' = 000741
6$:     MOV     #2$,R1          ;LOAD EXPECTED SR2 CONTENTS INTO R1
        CMP     R1,WASSR2       ;WAS SR2 LOCKED BY PLF ABORT?
        BEQ     7$
        EMT                     ;SR2 DIDN'T LOCK UP VIRTUAL ADDRESS
                                ;FOR TIGHTER SCOPE LOOP
                                ;REPLACE ERROR CALL WITH
                                ;'BR 2$'' = 000741
7$:     BIC     #160000,SR0     ;CLEAR ERROR BITS THAT WERE SET IN SR0
        MOV     #T04,@#4        ;RESTORE ADDRESS OF NORMAL CPU TRAP HANDLER
        MOV     #T0250,@#250    ;RESTORE ADDRESS OF NORMAL M.M. TRAP HANDLER
```

```
10791  045432  012767  077406  124656          MOV     #77406,KIPDR7   ;REMAP PAGE 7 TO READ/WRITE PLF=177
10792
10793
10794                                  ;*****************************************************************************
10795                                  ;TEST 412       PC & PSW SAVED FOR KT ERROR DURING SERVICE OF TIMEOUT ERROR
10796                                  ;*****************************************************************************
10797  045440                          TS412:
10798  045440  004767  002260          1$:     JSR     PC,TOFF         ;TURN T-BIT TRAPPING OFF FOR THIS TEST
10799  045444  012767  001400  132174          MOV     #1400,UIPAR3    ;MAP USER PAGE 3 TO 24-28K
10800  045452  012767  001400  132170          MOV     #1400,UIPAR4    ;MAP USER PAGE 4 TO 24-28K
10801  045460  012767  077402  132120          MOV     #77402,UIPDR3   ;MAP USER PAGE 3 READ-ONLY
10802  045466  012767  077406  132114          MOV     #77406,UIPDR4   ;MAP USER PAGE 4 READ/WRITE
10803  045474  012737  045542  000004          MOV     #4$,@#4         ;LOAD ADDRESS OF 4$ IN CPU (TIMEOUT) VECTOR
10804  045502  012737  140017  000006          MOV     #140017,@#6     ;LOAD PSW THAT SHOULD BE PUT ON STACK IN VECTOR+2
10805  045510  012737  045542  000250          MOV     #4$,@#250       ;LOAD ADDRESS OF 4$ IN M.M. TRAP VECTOR
10806  045516  012737  000340  000252          MOV     #340,@#252      ;LOAD A KERNEL PSW IN MMVEC+2
10807  045524  012767  140000  132244  2$:     MOV     #140000,PSW     ;GO TO USER MODE
10808  045532  012706  100002                  MOV     #100002,USP     ;SET USER STACK PTR. SO SECOND PUSH IS IN PG. 3
10809  045536  005737  177700          3$:     TST     @#177700        ;CAUSE TIMEOUT ERROR THAT WILL CAUSE
10810                                                                  ;R/O ERROR WHEN TRY TO SAVE OLD PC
10811  045542  016601  000002          4$:     MOV     2(KSP),R1       ;PUT PSW SAVED ON KERNEL STACK INTO R1
10812  045546  011603                          MOV     (KSP),R3        ;PUT PC SAVED ON KERNEL STACK INTO R3
10813  045550  016767  132016  171454          MOV     SR0,WASSR0      ;READ THE CONTENTS OF M.M. STATUS REG. 0
10814  045556  016767  132014  171450          MOV     SR2,WASSR2      ;READ THE CONTENTS OF M.M. STATUS REG. 2
10815  045564  042767  160000  132000          BIC     #160000,SR0     ;CLEAR THE ERROR BITS IN SR0
10816  045572  005067  132200                  CLR     PSW             ;BE SURE IN KERNEL MODE
10817  045576  012706  001000                  MOV     #KERSTK,KSP     ;RESTORE KERNEL STACK POINTER
10818  045602  012767  140000  132166          MOV     #140000,PSW     ;GO TO USER MODE
10819  045610  012706  000600                  MOV     #USESTK,USP     ;RESTORE USER STACK POINTER
10820  045614  005067  132156                  CLR     PSW             ;GO BACK TO KERNEL MODE
10821  045620  005067  171414                  CLR     $TMP0           ;CLEAR ERROR INDICATOR
10822  045624  020127  170017                  CMP     R1,#170017      ;WAS THE PSW SAVED THE ONE PICKED UP BY THE
10823                                                                  ;TIMEOUT TRAP FROM ERRVEC+2?
10824                                                                  ;VALUE 170017 = PSW FROM LOC. 6 WITH
10825                                                                  ;PREVIOUS MODE BITS = USER
10826  045630  001402                          BEQ     5$              ;BRANCH IF YES
10827  045632  005267  171402                  INC     $TMP0           ;WRONG PSW SAVED DURING "DOUBLE ERROR" SEQUENCE
10828  045636  020327  045542          5$:     CMP     R3,#3$+4        ;WAS THE PC AT THE TIME OF THE TIMEOUT ERROR
10829                                                                  ;SAVED ON THE STACK?
10830  045642  001402                          BEQ     6$              ;BRANCH IF YES
10831  045644  005267  171370                  INC     $TMP0           ;WRONG PC SAVED DURING TRAP SEQUENCE
10832  045650  026727  171356  020147  6$:     CMP     WASSR0,#20147   ;DID SR0 REPORT - USER, PAGE 3, R/O ABORT?
10833  045656  001402                          BEQ     7$              ;BRANCH IF YES
10834  045660  005267  171354                  INC     $TMP0           ;SR0 DID NOT REPORT R/O ABORT
10835  045664  026727  171344  045536  7$:     CMP     WASSR2,#3$      ;DID SR2 LOCK UP VIRTUAL ADDR. OF LAST
10836                                                                  ;INSTRUCTION SUCCESSFULLY FETCHED?
10837  045672  001402                          BEQ     8$              ;BRANCH IF YES
10838  045674  005267  171340                  INC     $TMP0           ;SR2 DID NOT LOCK UP ADDR. OF TIMEOUT INST.
10839  045700  005767  171334          8$:     TST     $TMP0           ;ANY "ERRORS" DURING TRAP SEQUENCE?
10840  045704  001401                          BEQ     9$
10841  045706  104000                          EMT
10842                                                                  ;THE WRONG PC OR PSW WERE SAVED
10843                                                                  ;OR SR0 OR SR2 DID NOT REPORT R/O
10844                                                                  ;ERROR DURING TIMEOUT - KT TRAP
10845                                                                  ;SEQUENCE
10846                                                                  ;FOR TIGHTER SCOPE LOOP
                                                                       ;REPLACE ERROR CALL WITH
```

```
10847                                                   ;'BR 2$'' = 000710
10848  045710  012737  021216  000004  9$:     MOV     #T04,@#4        ;RESTORE ADDRESS OF NORMAL CPU TRAP HANDLER
10849  045716  012737  000340  000006          MOV     #340,@#6        ;RELOAD ERRVEC+2 WITH KERNEL PSW
10850  045724  012737  021236  000250          MOV     #T0250,@#250    ;RESTORE ADDRESS OF NORMAL M.M. TRAP HANDLER
10851  045732  012767  077406  131646          MOV     #77406,UIPDR3   ;REMAP USER PAGE 3 READ/WRITE
10852  045740  004767  002014                  JSR     PC,TON          ;TURN T-BIT TRAPPING BACK ON
10853                                          ;*****************************************************************************
10854                                          ;*
10855                                          ;*      THIS GROUP OF TESTS WILL TEST ALL THE LOGIC ASSOCIATED WITH
10856                                          ;*      THE 'MOVE FROM PREVIOUS'' AND MOVE TO PREVIOUS'' INSTRUCTIONS.
10857                                          ;*
10858                                          ;*****************************************************************************
10859
10860                                          ;*****************************************************************************
10861                                          ;TEST 413       MOVE FROM PREVIOUS (USER) I-SPACE
10862                                          ;*****************************************************************************
10863                                          TS413:
10864  045744                                  1$:     CLR     KIPAR0          ;MAP KERNEL PAGE 0 TO 0-4K
10865  045744  005067  124370                          MOV     #200,KIPAR1     ;MAP KERNEL PAGE 1 TO 4-8K
10866  045750  012767  000200  124364                  MOV     #400,KIPAR2     ;MAP KERNEL PAGE 2 TO 8-12K
10867  045756  012767  000400  124360                  MOV     #600,KIPAR3     ;MAP KERNEL PAGE 3 TO 12-16K
10868  045764  012767  000600  124354                  MOV     #1400,KIPAR4    ;MAP KERNEL PAGE 4 TO 24-28K
10869  045772  012767  001400  124350                  MOV     #7600,KIPAR7    ;MAP KERNEL PAGE 7 TO THE I/O PAGE
10870  046000  012767  007600  124350                  MOV     #77406,R0       ;MAKE ALL KERNEL I-SPACE PAGES RESIDENT
10871  046006  012700  077406                                                  ;READ/WRITE, LENGTH 200 BLOCKS
10872
10873  046012  012702  000010                          MOV     #10,R2          ;SET LOOP COUNTER TO 8
10874  046016  012701  172300                          MOV     #KIPDR0,R1      ;PUT ADDRESS OF FIRST PDR IN R1
10875  046022  010021                          2$:     MOV     R0,(R1)+        ;LOAD PDR WITH 77406
10876  046024  077202                                  SOB     R2,2$           ;LOOP TO 2$ UNTIL ALL PDRS LOADED
10877  046026  012702  000010                          MOV     #10,R2          ;SET LOOP COUNTER TO 8
10878  046032  012701  177600                          MOV     #UIPDR0,R1      ;PUT ADDRESS OF FIRST PDR IN R1
10879  046036  010021                          3$:     MOV     R0,(R1)+        ;LOAD PDR WITH 77406
10880  046040  077202                                  SOB     R2,3$           ;LOOP TO 3$ UNTIL ALL PDRS LOADED
10881  046042  012767  000000  131570                  MOV     #000,UIPAR0     ;MAP USER I PAGE 0 TO 0-4K
10882  046050  012767  000200  131564                  MOV     #200,UIPAR1     ;MAP USER I PAGE 1 TO 4-8K
10883  046056  012767  000400  131560                  MOV     #400,UIPAR2     ;MAP USER I PAGE 2 TO 8-12K
10884  046064  012767  000600  131554                  MOV     #600,UIPAR3     ;MAP USER I PAGE 3 TO 12-16K
10885  046072  012767  007600  131556                  MOV     #7600,UIPAR7    ;MAP USER I PAGE 7 TO THE I/O PAGE
10886  046100                                  4$:
10887  046100  012767  077406  124202                  MOV     #77406,KIPDR4   ;KERNEL I-SPACE PAGE 4 READ/WRITE
10888  046106  012767  001400  124234                  MOV     #1400,KIPAR4    ;MAP KERNEL I PAGE 4 TO 24K
10889  046114  012767  001400  131526                  MOV     #1400,UIPAR4    ;MAP USER I PAGE 4 TO 24K
10890  046122  012700  036514                          MOV     #36514,R0       ;LOAD DATA PATTERN INTO R0
10891  046126  010037  100000                          MOV     R0,@#100000     ;LOAD DATA PATTERN INTO PHY 140000
10892  046132  012767  046430  132110                  MOV     #23$,MMVEC      ;SET M.M. VECTOR TO 23$
10893  046140  105067  124144                          CLRB    KIPDR4          ;MAKE KERNEL I-SPACE PAGE 4 NON-RESIDENT
10894                                                  ;THE FOLLOWING WILL TEST DSTM=0 MFPI
10895                                                  ;
10896  046144  012767  030340  131624  5$:     MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10897  046152  006506                    6$:     MFPI    USP             ;PUT USER STACK POINTER ON KERNEL
10898                                                                  ;STACK
10899  046154  022706  001000                          CMP     #KERSTK,KSP     ;WAS SOMETHING PUSHED ON STACK AT 6$
10900  046160  001405                                  BEQ     7$              ;BRANCH IF NOTHING WAS PUSHED
10901  046162  012600                                  MOV     (KSP)+,R0       ;POP KERNEL STACK INTO R0
10902  046164  012701  000600                          MOV     #USESTK,R1      ;EXPECTING TO GET 700 AS USP
```

```
10903  046170  020001                          CMP     R0,R1           ;DID YOU GET THE RIGHT POINTER?
10904  046172  001401                          BEQ     8$
10905  046174                          7$:
10906  046174  104000                          EMT                     ;WRONG THING WAS PUSHED ON STACK
10907                                                                  ;FOR TIGHTER SCOPE LOOP
10908                                                                  ;REPLACE ERROR CALL WITH
10909                                                                  ;'BR 5$'' = 000763
10910  046176                          8$:     ;THE FOLLOWING WILL TEST DSTM=1 MFPI.
10911  046176  012700  036514                  MOV     #36514,R0       ;RELOAD DATA PATTERN IN R0
10912  046202  012767  030340  131566  9$:     MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10913  046210  012702  100000                  MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
10914  046214  006512                          MFPI    (R2)            ;READ FROM PHYSICAL 140000
10915  046216  012601                          MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
10916  046220  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
10917  046222  001401                          BEQ     10$
10918  046224  104000                          EMT                     ;WRONG DATA WAS FETCHED
10919                                                                  ;FOR TIGHTER SCOPE LOOP
10920                                                                  ;REPLACE ERROR CALL WITH
10921                                                                  ;'BR 9$'' = 000766
10922  046226                          10$:    ;THE FOLLOWING WILL TEST DSTM=2 MFPI.
10923  046226  012767  030340  131542  11$:    MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10924  046234  012702  100000                  MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
10925  046240  006522                          MFPI    (R2)+           ;READ FROM PHYSICAL 140000
10926  046242  012601                          MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
10927  046244  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
10928  046246  001401                          BEQ     12$
10929  046250  104000                          EMT                     ;WRONG DATA WAS FETCHED
10930                                                                  ;FOR TIGHTER SCOPE LOOP
10931                                                                  ;REPLACE ERROR CALL WITH
10932                                                                  ;'BR 11$'' = 000766
10933  046252                          12$:    ;THE FOLLOWING WILL TEST DSTM=3 MFPI.
10934  046252  012767  030340  131516  13$:    MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10935  046260  006537  100000                  MFPI    @#100000        ;READ FROM PHYSICAL 140000
10936  046264  012601                          MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
10937  046266  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
10938  046270  001401                          BEQ     14$
10939  046272  104000                          EMT                     ;WRONG DATA WAS FETCHED
10940                                                                  ;FOR TIGHTER SCOPE LOOP
10941                                                                  ;REPLACE ERROR CALL WITH
10942                                                                  ;'BR 13$'' = 000767
10943  046274                          14$:    ;THE FOLLOWING WILL TEST DSTM=4 MFPI.
10944  046274  012767  030340  131474  15$:    MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10945  046302  012702  100002                  MOV     #100002,R2      ;LOAD VIRTUAL ADDRESS INTO R2
10946  046306  006542                          MFPI    -(R2)           ;READ FROM PHYSICAL 140000
10947  046310  012601                          MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
10948  046312  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
10949  046314  001401                          BEQ     16$
10950  046316  104000                          EMT                     ;WRONG DATA WAS FETCHED
10951                                                                  ;FOR TIGHTER SCOPE LOOP
10952                                                                  ;REPLACE ERROR CALL WITH
10953                                                                  ;'BR 15$'' = 000766
10954  046320                          16$:
10955                                  ;THE FOLLOWING WILL TEST DSTM=5 MFPI.
10956                                  ;
10957  046320  012767  030340  131450  17$:    MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10958  046326  012767  100000  170710          MOV     #100000,$TMP2   ;LOAD TEST LOC. VIRT. ADDR INTO LOC. $TMP2
```

```
10959  046334  012702  037246              MOV     #<$TMP2+2>,R2   ;LOAD ADDR. OF $TMP2+2 INTO R2
10960  046340  006552                      MFPI    @-(R2)          ;READ FROM PHYSICAL 140000
10961  046342  012601                      MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
10962  046344  020001                      CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
10963  046346  001401                      BEQ     18$
10964  046350  104000                      EMT                     ;WRONG DATA WAS FETCHED
10965                                                              ;FOR TIGHTER SCOPE LOOP
10966                                                              ;REPLACE ERROR CALL WITH
10967                                                              ;'BR 17$'' = 000763
10968  046352                      18$:    ;THE FOLLOWING WILL TEST DSTM=6 MFPI.
10969                                      ;
10970  046352  012767  030340  131416 19$:  MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10971  046360  005002                      CLR     R2              ;MAKE REGISTER 2 A ZERO
10972  046362  006562  100000              MFPI    100000(R2)      ;READ FROM PHYSICAL 140000
10973  046366  012601                      MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
10974  046370  020001                      CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
10975  046372  001401                      BEQ     20$
10976  046374  104000                      EMT                     ;WRONG DATA WAS FETCHED
10977                                                              ;FOR TIGHTER SCOPE LOOP
10978                                                              ;REPLACE ERROR CALL WITH
10979                                                              ;'BR 19$'' = 000766
10980  046376                      20$:    ;THE FOLLOWING WILL TEST DSTM=7 MFPI.
10981                                      ;
10982  046376  012767  030340  131372 21$:  MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
10983  046404  012767  100000  170632      MOV     #100000,$TMP2   ;LOAD TEST LOC. V.A. INTO $TMP2
10984  046412  012702  037244              MOV     #$TMP2,R2       ;LOAD ADDRESS OF $TMP2 INTO R2
10985  046416  006572  000000              MFPI    @0(R2)          ;USE $TMP2 TO FETCH VIRTUAL
10986                                                              ;ADDRESS OF 140000
10987  046422  012601                      MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
10988  046424  020001                      CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
10989  046426  001401                      BEQ     22$
10990  046430                      23$:
10991  046430  104000                      EMT                     ;WRONG DATA WAS FETCHED
10992                                                              ;FOR TIGHTER SCOPE LOOP
```

```
10993                                                          ;REPLACE ERROR CALL WITH
10994                                                          ;'BR 21$'' = 000762
10995  046432  012767  021236  131610  22$:   MOV    #T0250,MMVEC    ;SET M.M. VECTOR TO NORMAL ROUTINE
10996
10997
```

```
10998                                   ;*****************************************************************************
10999                                   ;TEST 414       MOVE TO PREVIOUS (USER) I-SPACE
11000                                   ;*****************************************************************************
11001  046440                           TS414:
11002  046440  012767  077406  123642  1$:     MOV     #77406,KIPDR4   ;KERNEL I-SPACE PAGE 4 READ/WRITE
11003  046446  012767  077406  131134          MOV     #77406,UIPDR4   ;USER I-SPACE PAGE 4 READ/WRITE
11004  046454  012767  001400  123666          MOV     #1400,KIPAR4    ;MAP KERNEL I PAGE 4 TO 24K
11005  046462  012767  001400  131160          MOV     #1400,UIPAR4    ;MAP USER I PAGE 4 TO 24K
11006  046470  012767  047162  131552          MOV     #20$,MMVEC      ;SET M.M. VECTOR TO 20$
11007                                          ;THE FOLLOWING WILL TEST DSTM=0 MTPI
11008                                          ;
11009  046476  012767  030340  131272  2$:     MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11010  046504  012746  007777                  MOV     #7777,-(KSP)    ;PUSH DATA ON KERNEL STACK
11011  046510  006606                          MTPI    USP             ;LOAD USER STACK POINTER
11012  046512  006506                          MFPI    USP             ;READ USER STACK POINTER
11013  046514  012601                          MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
11014  046516  022701  007777                  CMP     #7777,R1        ;WAS USER STACK POINTER CHANGED
11015  046522  001401                          BEQ     3$
11016  046524  104000                          EMT                     ;USER STACK POINTER NOT CHANGED
11017                                                                  ;FOR TIGHTER SCOPE LOOP
11018                                                                  ;REPLACE ERROR CALL WITH
11019                                                                  ;'BR 2$'' = 000764
11020  046526  012767  030340  131242  3$:     MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11021  046534  012746  000600                  MOV     #USESTK,-(KSP)  ;GET READY TO RESTORE USER S. POINT
11022  046540  006606                          MTPI    USP             ;RESTORE USER STACK POINTER
11023  046542                           4$:     ;THIS WILL TEST DSTM = 1 MTPI.
11024  046542  012702  100000                  MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
11025  046546  012700  125252                  MOV     #125252,R0      ;LOAD TEST DATA INTO R0
11026  046552  010046                   5$:     MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
11027  046554  105067  123530                  CLRB    KIPDR4          ;MAKE KERNEL I PAGE 4 NON-RESIDENT
11028  046560  006612                          MTPI    (R2)            ;LOAD TEST DATA INTO PHYSICAL 140000
11029  046562  112767  000006  123520          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
11030  046570  011201                          MOV     (R2),R1         ;READ FROM ADDRESS 140000
11031  046572  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED AT CORRECT PLACE
11032  046574  001401                          BEQ     6$
11033  046576  104000                          EMT                     ;INCORRECT STORE
11034                                                                  ;FOR TIGHTER SCOPE LOOP
11035                                                                  ;REPLACE ERROR CALL WITH
11036                                                                  ;'BR 5$'' = 000765
11037  046600                           6$:     ;THE FOLLOWING WILL TEST DSTM=2 MTPI.
11038                                          ;
11039  046600  012767  030340  131170          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11040  046606  012700  125252                  MOV     #125252,R0      ;LOAD TEST DATA INTO R0
11041  046612  012702  100000                  MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
11042  046616  010046                   8$:     MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
11043  046620  105067  123464                  CLRB    KIPDR4          ;MAKE KERNEL PAGE 4 NON-RESIDENT
11044  046624  006612                          MTPI    (R2)            ;LOAD TEST DATA INTO PHYSICAL 140000
11045  046626  112767  000006  123454          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESISENT
11046  046634  013701  100000                  MOV     @#100000,R1     ;READ FROM ADDRESS 140000
11047  046640  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
11048  046642  001401                          BEQ     9$
11049  046644  104000                          EMT                     ;INCORRECT STORE
11050                                                                  ;FOR TIGHTER SCOPE LOOP
11051                                                                  ;REPLACE ERROR CALL WITH
11052                                                                  ;'BR 8$'' = 000764
11053  046646                           9$:     ;THIS WILL TEST DSTM = 3 MTPI.
```

```
11054  046646  012767  030340  131122          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11055  046654  012700  052525                  MOV     #52525,R0       ;LOAD TEST DATA INTO R0
11056  046660  010046                  10$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
11057  046662  105067  123422                  CLRB    KIPDR4          ;MAKE KERNEL I PAGE 4 NON-RESIDENT
11058  046666  006637  100000                  MTPI    @#100000        ;LOAD TEST DATA INTO PHYSICAL 140000
11059  046672  112767  000006  123410          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
11060  046700  013701  100000                  MOV     @#100000,R1     ;READ FROM ADDRESS 140000
11061  046704  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
11062  046706  001531                          BEQ     TS415
11063  046710  104000                          EMT                     ;
11064  046712  001401                          BEQ     11$
11065  046714  104000                          EMT                     ;INCORRECT STORE
11066                                                                  ;FOR TIGHTER SCOPE LOOP
11067                                                                  ;REPLACE ERROR CALL WITH
11068                                                                  ;'BR 10$'' = 000763
11069  046716                           11$:   ;THIS WILL TEST DSTM = 4 MTPI.
11070  046716  012767  030340  131052          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11071  046724  012700  125252                  MOV     #125252,R0      ;LOAD TEST DATA INTO R0
11072  046730  010046                  12$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
11073  046732  012702  100002                  MOV     #100002,R2      ;LOAD VIRTUAL ADDRESS INTO R2
11074  046736  105067  123346                  CLRB    KIPDR4          ;MAKE KERNEL I PAGE 4 NON-RESIDENT
11075  046742  006642                          MTPI    -(R2)           ;LOAD TEST DATA INTO PHYSICAL 140000
11076  046744  112767  000006  123336          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
11077  046752  013701  100000                  MOV     @#100000,R1     ;READ FROM ADDRESS 140000
11078  046756  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
11079  046760  001401                          BEQ     13$
11080  046762  104000                          EMT                     ;INCORRECT STORE
11081                                                                  ;FOR TIGHTER SCOPE LOOP
11082                                                                  ;REPLACE ERROR CALL WITH
11083                                                                  ;'BR 12$'' = 000762
11084  046764                           13$:   ;THE FOLLOWING WILL TEST DSTM=5 MTPI.
11085                                          ;
11086  046764  012767  030340  131004          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11087  046772  012700  052525                  MOV     #52525,R0       ;LOAD TEST DATA INTO R0
11088  046776  012702  037246                  MOV     #<$TMP2+2>,R2   ;LOAD ADDR. OF LOC. $TMP2+2 INTO R2
11089  047002  012767  100000  170234          MOV     #100000,$TMP2   ;LOAD VIRT. ADDR. OF TEST LOC. INTO $TMP2
11090  047010  010046                  14$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
11091  047012  105067  123272                  CLRB    KIPDR4          ;MAKE KERNEL PAGE 4 NON-RESIDENT
11092  047016  006652                          MTPI    @-(R2)          ;LOAD TEST DATA INTO PHYSICAL 140000
11093  047020  112767  000006  123262          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
11094  047026  013701  100000                  MOV     @#100000,R1     ;READ FROM ADDRESS 140000
11095  047032  020001                          CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
11096  047034  001401                          BEQ     15$
11097  047036  104000                          EMT                     ;INCORRECT STORE
11098                                                                  ;FOR TIGHTER SCOPE LOOP
11099                                                                  ;REPLACE ERROR CALL WITH
11100                                                                  ;'BR 14$'' = 000764
11101  047040                           15$:   ;THIS WILL TEST DSTM = 6 MTPI.
11102                                          ;
11103  047040  012767  030340  130730          MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11104  047046  012700  052525                  MOV     #52525,R0       ;LOAD TEST DATA INTO R0
11105  047052  005002                          CLR     R2              ;MAKE REGISTER 2 ZERO
11106  047054  010046                  16$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
11107  047056  105067  123226                  CLRB    KIPDR4          ;MAKE KERNEL I PAGE 4 NON-RESIDENT
11108  047062  006662  100000                  MTPI    100000(R2)      ;LOAD TEST DATA INTO PHYSICAL 140000
11109  047066  112767  000006  123214          MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
```

```
11110  047074  013701  100000          MOV     @#100000,R1     ;READ FROM ADDRESS 140000
11111  047100  020001                  CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
11112  047102  001401                  BEQ     17$
11113  047104  104000                  EMT                     ;INCORRECT STORE
11114                                                          ;FOR TIGHTER SCOPE LOOP
11115                                                          ;REPLACE ERROR CALL WITH
11116                                                          ;'BR 16$'' = 000763
11117  047106                  17$:    ;THE FOLLOWING WILL TEST DSTM=7 MTPI.
11118                                  ;
11119  047106  012767  030340  130662  MOV     #030340,PSW     ;MAKE PREVIOUS MODE USER
11120  047114  012700  125252          MOV     #125252,R0      ;LOAD TEST DATA INTO R0
11121  047120  012767  100000  170116  MOV     #100000,$TMP2   ;LOAD VIRT. ADDR. OF TEST LOCATION
11122                                                          ;INTO LOCATION $TMP2
11123  047126  012702  037244          MOV     #$TMP2,R2       ;LOAD ADDRESS OF $TMP2 INTO R2
11124  047132  010046          18$:    MOV     R0,-(KSP)       ;PUSH TEST DATA ON KERNEL STACK
11125  047134  105067  123150          CLRB    KIPDR4          ;MAKE KERNEL PAGE 4 NON-RESIDENT
11126  047140  006672  000000          MTPI    @0(R2)          ;LOAD TEST DATA INTO PHYSICAL 140000
11127  047144  112767  000006  123136  MOVB    #006,KIPDR4     ;MAKE KERNEL PAGE 4 RESIDENT
11128  047152  013701  100000          MOV     @#100000,R1     ;READ FROM ADDRESS 140000
11129  047156  020001                  CMP     R0,R1           ;SEE IF DATA WAS STORED CORRECTLY
11130  047160  001401                  BEQ     19$
11131  047162                  20$:
11132  047162  104000                  EMT                     ;INCORRECT STORE
11133                                                          ;FOR TIGHTER SCOPE LOOP
11134                                                          ;REPLACE ERROR CALL WITH
11135                                                          ;'BR 18$'' = 000763
11136  047164  012767  021236  131056  19$:    MOV     #T0250,MMVEC    ;RESTORE M.M. VECTOR TO NORMAL ROUTINE
11137
11138
11139
11140                          ;*****************************************************************************************
11141                          ;TEST 415       MOVE FROM PREVIOUS (KERNEL) I-SPACE TO USER MODE
11142                          ;*****************************************************************************************
11143  047172                  TS415:
11144  047172  012700  077406  1$:     MOV     #77406,R0       ;MAKE ALL USER I-SPACE PAGES RESIDENT
11145                                                          ;READ/WRITE, LENGTH 200 BLOCKS
11146  047176  012702  000010          MOV     #10,R2          ;SET LOOP COUNTER TO 8
11147  047202  012701  177600          MOV     #UIPDR0,R1      ;LOAD ADDRESS OF FIRST PDR IN R1
11148  047206  010021          2$:     MOV     R0,(R1)+        ;LOAD PDR WITH 77406
11149  047210  077202                  SOB     R2,2$           ;LOOP UNTIL 8 USER PDRS LOADED
11150  047212  012737  140340  130556  3$:     MOV     #140340,PSW     ;GO TO USER MODE FOR THIS TEST
11151  047220  012767  077406  123062          MOV     #77406,KIPDR4   ;KERNEL I-SPACE PAGE 4 READ/WRITE
11152  047226  012767  001400  123114          MOV     #1400,KIPAR4    ;MAP KERNEL I PAGE 4 TO 24K
11153  047234  012767  001400  130406          MOV     #1400,UIPAR4    ;MAP USER I PAGE 4 TO 24K
11154  047242  012700  036514                  MOV     #36514,R0       ;LOAD DATA PATTERN INTO R0
11155  047246  010037  100000                  MOV     R0,@#100000     ;LOAD DATA PATTERN INTO PHY 140000
11156  047252  012702  100000                  MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
11157                                  ;THE FOLLOWING WILL TEST DSTM=0 MFPI
11158                                  ;
11159  047256  012767  047554  130764          MOV     #21$,MMVEC      ;SET M.M. VECTOR TO 21$
11160  047264  105067  130320                  CLRB    UIPDR4          ;MAKE USER I-SPACE PAGE 4 NON-RESIDENT
11161  047270  012767  140340  130500          MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
11162  047276  006506          4$:     MFPI    KSP             ;PUT KERNEL STACK POINTER ON USER STACK
11163  047300  022706  000600                  CMP     #USESTK,USP     ;WAS SOMETHING PUSHED ON STACK AT 1$
11164  047304  001405                  BEQ     5$              ;BRANCH IF NOTHING WAS PUSHED
11165  047306  012600                  MOV     (USP)+,R0       ;POP USER STACK INTO R0
```

```
11166  047310  012701  001000                  MOV     #KERSTK,R1      ;EXPECTING 1100 AS KSP
11167  047314  020001                          CMP     R0,R1           ;DID YOU GET THE RIGHT POINTER?
11168  047316  001401                          BEQ     6$
11169  047320                          5$:
11170  047320  104000                          EMT                     ;WRONG THING WAS PUSHED ON STACK
11171                                                                   ;FOR TIGHTER SCOPE LOOP
11172                                                                   ;REPLACE ERROR CALL WITH
11173                                                                   ;'BR 4$'' = 000766
11174  047322                          6$:     ;THE FOLLOWING WILL TEST DSTM=1 MFPI.
11175  047322  012767  140340  130446  7$:     MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
11176  047330  012700  036514                  MOV     #36514,R0       ;LOAD DATA EXPECTED INTO R0
11177  047334  012702  100000                  MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
11178  047340  006512                          MFPI    (R2)            ;READ FROM PHYSICAL 140000
11179  047342  012601                          MOV     (USP)+,R1       ;POP USER STACK INTO R1
11180  047344  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
11181  047346  001401                          BEQ     9$
11182  047350  104000                          EMT                     ;WRONG DATA WAS FETCHED
11183                                                                   ;FOR TIGHTER SCOPE LOOP
11184                                                                   ;REPLACE ERROR CALL WITH
11185                                                                   ;'BR 7$'' = 000764
11186                                          ;THE FOLLOWING WILL TEST DSM=2 MFPI.
11187  047352  012767  140340  130416  9$:     MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
11188  047360  012702  100000                  MOV     #100000,R2      ;LOAD VIRTUAL ADDRESS INTO R2
11189  047364  006522                          MFPI    (R2)+           ;READ FROM PHYSICAL 140000
11190  047366  012601                          MOV     (USP)+,R1       ;POP USER STACK INTO R1
11191  047370  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
11192  047372  001401                          BEQ     11$
11193  047374  104000                          EMT                     ;WRONG DATA WAS FETCHED
11194                                                                   ;FOR TIGHTER SCOPE LOOP
11195                                                                   ;REPLACE ERROR CALL WITH
11196                                                                   ;'BR 9$'' = 000766
11197                                          ;THE FOLLOWING WILL TEST DSTM=3 MFPI.
11198  047376  012767  140340  130372  11$:    MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
11199  047404  006537  100000                  MFPI    @#100000        ;READ FROM PHYSICAL 140000
11200  047410  012601                          MOV     (USP)+,R1       ;POP USER STACK INTO R1
11201  047412  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
11202  047414  001401                          BEQ     13$
11203  047416  104000                          EMT                     ;WRONG DATA WAS FETCHED
11204                                                                   ;FOR TIGHTER SCOPE LOOP
11205                                                                   ;REPLACE ERROR CALL WITH
11206                                                                   ;'BR 11$'' = 000767
11207                                          ;THE FOLLOWING WILL TEST DSTM=4 MFPI.
11208  047420  012767  140340  130350  13$:    MOV     #140340,PSW     ;MAKE PREVIOUS MODE DERNEL PRESENT USER
11209  047426  012702  100002                  MOV     #100002,R2      ;LOAD VIRTUAL ADDRESS INTO R2
11210  047432  006542                          MFPI    -(R2)           ;READ FROM PHYSICAL 140000
11211  047434  012601                          MOV     (USP)+,R1       ;POP USER STACK INTO R1
11212  047436  020001                          CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
11213  047440  001401                          BEQ     15$
11214  047442  104000                          EMT                     ;WRONG DATA WAS FETCHED
11215                                                                   ;FOR TIGHTER SCOPE LOOP
11216                                                                   ;REPLACE ERROR CALL WITH
11217                                                                   ;'BR 13$'' = 000766
11218                                          ;THE FOLLOWING WILL TEST DSTM=5 MFPI.
11219                                          ;
11220  047444  012767  140340  130324  15$:    MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
11221  047452  012767  100000  167564          MOV     #100000,$TMP2   ;LOAD TEST LOC. VIRT. ADDR INTO LOC. $TMP2
```

```
11222  047460  012702  037246          MOV     #<$TMP2+2>,R2   ;LOAD ADDRESS OF $TMP2+2 INTO R2
11223  047464  006552                  MFPI    @-(R2)          ;READ FROM PHYSICAL 140000
11224  047466  012601                  MOV     (USP)+,R1       ;POP USER STACK INTO R1
11225  047470  020001                  CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
11226  047472  001401                  BEQ     17$
11227  047474  104000                  EMT                     ;WRONG DATA WAS FETCHED
11228                                                          ;FOR TIGHTER SCOPE LOOP
11229                                                          ;REPLACE ERROR CALL WITH
11230                                                          ;'BR 15$'' = 000763
11231                                  ;THE FOLLOWING WILL TEST DSTM=6 MFPI.
11232                                  ;
11233  047476  012767  140340  130272 17$:    MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
11234  047504  005002                  CLR     R2              ;MAKE REGISTER 2 A ZERO
11235  047506  006562  100000          MFPI    100000(R2)      ;READ FROM PHYSICAL 140000
11236  047512  012601                  MOV     (USP)+,R1       ;POP USER STACK INTO R1
11237  047514  020001                  CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
11238  047516  001401                  BEQ     19$
11239  047520  104000                  EMT                     ;WRONG DATA WAS FETCHED
11240                                                          ;FOR TIGHTER SCOPE LOOP
11241                                                          ;REPLACE ERROR CALL WITH
11242                                                          ;'BR 17$'' = 000766
11243                                  ;THE FOLLOWING WILL TEST DSTM=7 MFPI.
11244                                  ;
11245  047522  012767  140340  130246 19$:    MOV     #140340,PSW     ;MAKE PREVIOUS MODE KERNEL PRESENT USER
11246  047530  012767  100000  167506          MOV     #100000,$TMP2   ;LOAD TEST LOC. VIRT. ADDR. INTO $TMP2
11247  047536  012702  037244          MOV     #$TMP2,R2       ;LOAD ADDRESS OF $TMP2 INTO R2
11248  047542  006572  000000          MFPI    @0(R2)          ;READ FROM PHYSICAL 140000
11249  047546  012601                  MOV     (USP)+,R1       ;POP USER STACK INTO R1
11250  047550  020001                  CMP     R0,R1           ;WAS DATA FETCHED SAME AS STORED
11251  047552  001401                  BEQ     20$
11252  047554                  21$:
11253  047554  104000                  EMT                     ;WRONG DATA WAS FETCHED
11254                                                          ;FOR TIGHTER SCOPE LOOP
11255                                                          ;REPLACE ERROR CALL WITH
11256                                                          ;'BR 19$'' = 000762
11257  047556  012767  021236  130464 20$:    MOV     #T0250,MMVEC    ;SET M.M. VECTOR TO NORMAL ROUTINE
11258  047564  012767  000340  130204          MOV     #00340,PSW      ;GO BACK TO KERNEL MODE, PREVIOUS KERNEL
11259
11260                                  ;*****************************************************************************************
11261                                  ;TEST 416       MOVE FROM/TO D-SPACE = MOVE FROM/TO I-SPACE
11262                                  ;*****************************************************************************************
11263  047572                  TS416:
11264  047572  012767  030340  130176 1$:     MOV     #030340,PSW     ;MAKE PREVIOUS MODE=USER,CURRENT=KERNEL
11265  047600  106506                  MFPD    USP             ;MFPD SHOULD ACT LIKE MFPI PUTTING
11266                                                          ;USER STACK POINTER ON THE KERNEL STACK
11267  047602  022706  001000          CMP     #KERSTK,KSP     ;WAS SOMETHING PUSHED ON KERNEL STACK?
11268  047606  001405                  BEQ     2$              ;BRANCH IF NO
11269  047610  012600                  MOV     (KSP)+,R0       ;POP KERNEL STACK INTO R0
11270  047612  012701  000600          MOV     #USESTK,R1      ;EXPECTING TO GET 700 AS USP
11271  047616  020001                  CMP     R0,R1           ;DID GET RIGHT POINTER VALUE?
11272  047620  001401                  BEQ     4$
11273  047622                  2$:
11274  047622  104000                  EMT                     ;WRONG THING WAS PUSHED ON STACK
11275                                                          ;FOR TIGHTER SCOPE LOOP
11276                                                          ;REPLACE ERROR CALL WITH
11277                                                          ;'BR 1$'' = 000763
```

```
11278  047624  012746  007777          4$:     MOV     #7777,-(KSP)    ;PUSH DATA ON KERNEL STACK
11279  047630  106606                          MTPD    USP             ;LOAD THE USER STACK POINTER
11280  047632  106506                          MFPD    USP             ;READ USER STACK POINTER
11281  047634  012601                          MOV     (KSP)+,R1       ;POP KERNEL STACK INTO R1
11282  047636  022701  007777                  CMP     #7777,R1        ;WAS USER STACK POINTER CHANGED?
11283  047642  001401                          BEQ     5$
11284  047644  104000                          EMT                     ;USER STACK POINTER NOT CHANGED
11285                                                                  ;FOR TIGHTER SCOPE LOOP
11286                                                                  ;REPLACE ERROR CALL WITH
11287                                                                  ;'BR 4$'' = 000767
11288  047646  012746  000600          5$:     MOV     #USESTK,-(KSP)  ;GET READY TO RESTORE USER STK. PTR.
11289  047652  106606                          MTPD    USP             ;RESTORE USER STACK POINTER
11290
11291                                  ;*****************************************************************************
11292                                  :TEST 417       MOVE FROM PREVIOUS I=SPACE (PREVIOUS=CURRENT=KERNEL)
11293                                  ;*****************************************************************************
11294  047654                          TS417:
11295  047654  005037  177776          1$:     CLR     @#PSW           ;SET PREVIOUS = CURRENT = KERNEL
11296  047660  012700  001000                  MOV     #KERSTK,R0      ;SETUP VALUE FOR STACK POINTER
11297  047664  010006                          MOV     R0,KSP          ;LOAD STACK POINTER
11298  047666  006506                          MFPI    KSP             ;THE VALUE ''STACK'' SHOULD BE PUSHED
11299                                                                  ;BEFORE BEING DECREMENTED
11300  047670  011601                          MOV     (KSP),R1        ;READ DATA WHICH WAS PUSHED
11301  047672  020001                          CMP     R0,R1           ;WAS THE ORIGINAL VALUE OF THE
11302                                                                  ;STACK POINTER PUSHED?
11303  047674  001401                          BEQ     2$
11304  047676  104000                          EMT                     ;MFPI FETCHED WRONG DATA
11305                                                                  ;FOR TIGHTER SCOPE LOOP
11306                                                                  ;REPLACE ERROR CALL WITH
11307                                                                  ;'BR 1$'' = 000766
11308  047700  005740                  2$:     TST     -(R0)           ;SETUP EXPECTED STACK POINTER VALUE
11309  047702  020600                          CMP     KSP,R0          ;WAS THE STACK POINTER DECREMENTED?
11310  047704  001401                          BEQ     3$
11311  047706  104000                          EMT                     ;STACK NOT PUSHED BY THE MFPI
11312                                                                  ;FOR TIGHTER SCOPE LOOP
11313                                                                  ;REPLACE ERROR CALL WITH
11314                                                                  ;'BR 1$'' = 000762
11315  047710  012706  001000          3$:     MOV     #KERSTK,KSP     ;RESTORE STACK POINTER
11316  047714  005067  127652                  CLR     SR0             ;TURN OFF MEMORY MANAGEMENT UNIT
11317  047720  000167  000404                  JMP     FPSTRT          ;GET OVER SUBROUTINES TO FLOATING POINT TESTS
```

```
11318                                   .SBTTL  ***** SUBROUTINES USED BY THIS PROGRAM *****
11319
11320                                   .SBTTL  TURN OFF T-BIT AND SAVE CURRENT PSW
11321                                   ;*******************************************************************************
11322                                   ;*
11323                                   ;*      THIS SUBROUTINE IS USED TO TURN OFF THE TRACE TRAP BIT IN THE PSW
11324                                   ;*      IF IT IS ON.  THE PROCESSOR STATUS IS SAVED IN ''TBITPS'' SO THAT
11325                                   ;*      THE PSW CAN BE RESTORED TO ITS PREVIOUS CONDITION WHEN CONDITIONS
11326                                   ;*      WARRANT T-BIT TRAPPING.
11327                                   ;*
11328                                   ;*******************************************************************************
11329  047724  036727  130046  000020  TOFF:   BIT     PSW,#TBIT       ;IS THE T-BIT SET IN THE PSW?
11330  047732  001411                   BEQ     1$              ;EXIT IF NO
11331  047734  016746  130036          MOV     PSW,-(SP)       ;PUSH PRESENT PSW ON THE STACK
11332  047740  011667  167272          MOV     (SP),TBITPS     ;ALSO SAVE IT IN ''TBITPS'' FOR
11333                                                          ;RESTORING LATER
11334  047744  042716  000020          BIC     #TBIT,(SP)      ;CLEAR THE T-BIT (BIT 4) IN THE PSW
11335  047750  012746  047756          MOV     #1$,-(SP)       ;PUSH PC OF ''RTS'' ON STACK
11336  047754  000006                   RTT                     ;''RETURN'' TO 1$ WITH T-BIT OFF
11337  047756  000207          1$:     RTS     PC              ;RETURN TO PROGRAM
11338
11339                                   .SBTTL  TURN ON T-BIT AND RESTORE PREVIOUS PSW
11340                                   ;*******************************************************************************
11341                                   ;*
11342                                   ;*      THIS SUBROUTINE IS USED TO RESTORE THE PROCESSOR STATUS TO ITS
11343                                   ;*      PREVIOUS CONDITION BY RESTORING THE ''T-BIT PSW'' SAVED BY THE
11344                                   ;*      ''TOFF'' SUBROUTINE IN THE ''TBITPS'' LOCATION.
11345                                   ;*
11346                                   ;*******************************************************************************
11347  047760  036727  167252  000020  TON:    BIT     TBITPS,#TBIT    ;WAS T-BIT ON IN THE PREVIOUS PSW?
11348  047766  001410                   BEQ     1$              ;EXIT IF NO
11349  047770  016746  167242          MOV     TBITPS,-(SP)    ;PUSH PREVIOUS PSW ON THE STACK
11350  047774  012767  000340  167234  MOV     #340,TBITPS     ;RESET THE ''TBITPS'' LOCATION
11351  050002  012746  050010          MOV     #1$,-(SP)       ;PUSH PC OF ''RTS'' ON STACK
11352  050006  000006                   RTT                     ;''RETURN'' TO 1$ WITH T-BIT RESTORED
11353  050010  000207          1$:     RTS     PC              ;RETURN TO PROGRAM
11354
11355                                   .SBTTL  SET ALL WRITEABLE BITS IN ALL PAR/PDR'S
11356                                   ;*******************************************************************************
11357                                   ;*
11358                                   ;*      THIS SUBROUTINE IS USED BY THE PAR/PDR DUAL ADDRESSING TEST
11359                                   ;*      TO SET ALL WRITEABLE BITS IN ALL KERNEL AND USE PAR'S AND
11360                                   ;*      PDR'S TO A 1.  THE ''INITIAL STATE'' OF HAVING ALL BITS=1 IS
11361                                   ;*      USED TO SEE THAT ONLY ONE REGISTER IS CLEARED IN RESPONSE TO
11362                                   ;*      A SINGLE PAR OR PDR ADDRESS.
11363                                   ;*
11364                                   ;*******************************************************************************
11365
11366  050012  012702  000010  SETREG: MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
11367  050016  012701  172300          MOV     #KIPDR0,R1      ;LOAD ADDRESS OF FIRST PDR INTO R1
11368  050022  012721  177777  1$:     MOV     #-1,(R1)+       ;SET BITS IN KERNEL PDR TO 1
11369  050026  077203                   SOB     R2,1$           ;LOOP TO 1$ UNTIL ALL KERNEL PDR'S LOADED
11370  050030  012702  000010          MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
11371  050034  012701  172340          MOV     #KIPAR0,R1      ;LOAD ADDRESS OF FIRST PAR INTO R1
11372  050040  012721  177777  2$:     MOV     #-1,(R1)+       ;SET BITS IN A KERNEL PAR TO 1
11373  050044  077203                   SOB     R2,2$           ;LOOP TO 2$ UNTIL ALL KERNEL PAR'S LOADED
```

```
11374  050046  012702  000010          MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
11375  050052  012701  177600          MOV     #UIPDR0,R1      ;LOAD ADDRESS OF FIRST PDR INTO R1
11376  050056  012721  177777  3$:     MOV     #-1,(R1)+       ;SET BITS IN A USER PDR TO 1
11377  050062  077203                  SOB     R2,3$           ;LOOP TO 3$ UNTIL ALL USER PDR'S LOADED
11378  050064  012702  000010          MOV     #10,R2          ;LOAD LOOP COUNTER WITH AN 8
11379  050070  012701  177640          MOV     #UIPAR0,R1      ;LOAD ADDRESS OF FIRST PAR INTO R1
11380  050074  012721  177777  4$:     MOV     #-1,(R1)+       ;SET BITS IN A USER PAR TO 1
11381  050100  077203                  SOB     R2,4$           ;LOOP TO 4$ UNTIL ALL USER PAR'S LOADED
11382  050102  000207                  RTS     PC              ;RETURN TO TEST
11383
11384                                  .SBTTL  READ & COMPARE KERNEL & USER PAR/PDR'S
11385                                  ;*****************************************************************************
11386                                  ;*
11387                                  ;*      THIS SUBROUTINE IS USED BY PAR/PDR DUAL ADDRESSING TEST TO
11388                                  ;*      READ ALL THE PAR'S AND PDR'S TO SEE THAT ONLY ONE REGISTER
11389                                  ;*      WAS CLEARED IN RESPONSE TO A SINGLE PAR OR PDR ADDRESS.
11390                                  ;*      ANY FAILURES FOUND BY THE PAR/PDR DUAL ADDRESSING TEST WILL
11391                                  ;*      BE REPORTED BY THIS SUBROUTINE.
11392                                  ;*
11393                                  ;*****************************************************************************
11394  050104                  CMPREG:
11395  050104  012701  172300          MOV     #KIPDR0,R1      ;LOAD ADDRESS OF FIRST KERNEL PDR IN R1
11396  050110  012704  000010          MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
11397  050114  012705  077416          MOV     #77416,R5       ;PUT EXPECTED PDR CONTENTS IN R5
11398  050120  021105          1$:     CMP     (R1),R5         ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
11399  050122  001403                  BEQ     2$              ;BRANCH IF YES
11400  050124  020100                  CMP     R1,R0           ;WAS IT THE REG. THAT WAS CLEARED?
11401  050126  001401                  BEQ     2$
11402  050130  104000                  EMT                     ;A PDR WAS EFFECTED BY CLEARING A DIFFERENT PAR/PRD
11403                                                          ;FOR TIGHTER SCOPE LOOP
11404                                                          ;REPLACE ERROR CALL WITH
11405                                                          ;AN 'RTS PC' = 000207
11406  050132  062701  000002  2$:     ADD     #2,R1           ;FORM NEXT ADDRESS
11407  050136  077410                  SOB     R4,1$           ;LOOP TO 1$ UNTIL ALL KERNEL PDR'S CHECKED
11408  050140  012701  172340          MOV     #KIPAR0,R1      ;LOAD ADDRESS OF FIRST KERNEL PAR IN R1
11409  050144  012704  000010          MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
11410                                                          ;****F11 CHANGE**** FROM #7777 TO #177777
11411  050150  012705  177777          MOV     #177777,R5      ;PUT EXPECTED PAR CONTENTS IN R5
11412  050154  021105          3$:     CMP     (R1),R5         ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
11413  050156  001403                  BEQ     4$              ;BRANCH IF YES
11414  050160  020100                  CMP     R1,R0           ;WAS IT THE REG. THAT WAS CLEARED?
11415  050162  001401                  BEQ     4$
11416  050164  104000                  EMT                     ;A PAR WAS EFFECTED BY CLEARING A DIFFENENT PAR/PDR
11417                                                          ;FOR TIGHTER SCOPE LOOP
11418                                                          ;REPLACE ERROR CALL WITH
11419                                                          ;AN 'RTS PC' = 000207
11420  050166  062701  000002  4$:     ADD     #2,R1           ;FORM NEXT ADDRESS
11421  050172  077410                  SOB     R4,3$           ;LOOP TO 3$ UNTIL ALL KERNEL PAR'S CHECKED
11422  050174  012701  177600          MOV     #UIPDR0,R1      ;LOAD ADDRESS OF FIRST USER PDR IN R1
11423  050200  012704  000010          MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
11424  050204  012705  077416          MOV     #77416,R5       ;PUT EXPECTED PDR CONTENTS IN R5
11425  050210  021105          5$:     CMP     (R1),R5         ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
11426  050212  001403                  BEQ     6$              ;BRANCH IF YES
11427  050214  020100                  CMP     R1,R0           ;WAS IT THE REG. THAT WAS CLEARED?
11428  050216  001401                  BEQ     6$
11429  050220  104000                  EMT                     ;A PDR WAS EFFECTED BY CLEARING A DIFFERENT PAR/PDR
```

```
11430                                                   ;FOR TIGHTER SCOPE LOOP
11431                                                   ;REPLACE ERROR CALL WITH
11432                                                   ;AN 'RTS PC'' = 000207
11433  050222  062701  000002        6$:     ADD     #2,R1           ;FORM NEXT ADDRESS
11434  050226  077410                 SOB     R4,5$           ;LOOP TO 5$ UNTIL ALL USER PDR'S CHECKED
11435  050230  012701  177640        MOV     #UIPAR0,R1      ;LOAD ADDRESS OF FIRST USER PAR IN R1
11436  050234  012704  000010        MOV     #10,R4          ;LOAD LOOP COUNTER WITH AN 8
11437                                                   ;****F11 CHANGE**** FROM #7777 TO #177777
11438  050240  012705  177777        MOV     #177777,R5      ;PUT EXPECTED PAR CONTENTS IN R5
11439  050244  021105                 7$:     CMP     (R1),R5         ;ARE ALL WRITEABLE BITS SET AS EXPECTED?
11440  050246  001403                 BEQ     8$              ;BRANCH IF YES
11441  050250  020100                 CMP     R1,R0           ;WAS IT THE REG. THAT WAS CLEARED?
11442  050252  001401                 BEQ     8$
11443  050254  104000                 EMT                     ;A PAR WAS EFFECTED BY CLEARING A DIFFERENT PAR/PDR
11444                                                   ;FOR TIGHTER SCOPE LOOP
11445                                                   ;REPLACE ERROR CALL WITH
11446                                                   ;AN 'RTS PC'' = 000207
11447  050256  062701  000002        8$:     ADD     #2,R1           ;FORM NEXT ADDRESS
11448  050262  077410                 SOB     R4,7$           ;LOOP TO 7$ UNTIL ALL USER PAR'S CHECKED
11449  050264  000207                 RTS     PC              ;RETURN TO TEST
11450
11451                                   .SBTTL  INHIBIT 'RESETS' WHILE UNDER APT
11452                                   ;*******************************************************************************
11453                                   ;*
11454                                   ;*      THIS SUBROUTINE CONTROLS THE USAGE OF RESET INST'S WHILE
11455                                   ;*      RUNNING UNDER APT. RESETS ARE ALLOWED DURING THE FIRST
11456                                   ;*      PASS OF THE DIAGNOSTIC.
11457                                   ;*
11458  050266  126727  130526  000001  CHKAPT: CMPB    $ENV,#1         ;ARE WE RUNNING UNDER APT?
11459  050274  001003                         BNE     1$              ;NO BRANCH
11460  050276  005767  130504                TST     $PASS           ;IS THIS THE FIRST PASS?
11461  050302  001002                         BNE     RETA            ;NO BRANCH
11462  050304  062705  000002          1$:    ADD     #2,R5           ;BUMP RETURN ADDRESS FOR NORMAL TESTING
11463  050310  000205                 RETA:   RTS     R5              ;RETURN
11464
11465
11466                                   .SBTTL  ERROR ROUTINE FOR MEMORY MANAGEMENT TEST
11467                                   ;*******************************************************************************
11468                                   ;*      THIS IS THE ONLY ERROR REPORT FOR ALL THE MMU TESTS
11469                                   ;*******************************************************************************
11470
11471  050312  012737  000002  001002  ERROR3: MOV     #2,@#$FATAL                     ;SET UP FATAL ERROR NUMBER
11472  050320  012767  000001  130452          MOV     #1,$MSGTY                       ;SET FATAL ERROR FLAG
11473  050326  000777                 MMUHLT: BR      .               ;STAY IN LOOP
11474
11475
11476
11477
11478
11479
11480
11481
11482
11483
11484          000244                   FPVECT=244
11485                                           .SBTTL  FPP REGISTER DEFINITIONS
```

```
11486                  000000                  AC0     =%0
11487                  000001                  AC1     =%1
11488                  000002                  AC2     =%2
11489                  000003                  AC3     =%3
11490                  000004                  AC4     =%4
11491                  000005                  AC5     =%5
11492                  000006                  AC6     =%6
11493                  000007                  AC7     =%7
11494
11495
11496  050330  012706  001000          FPSTRT: MOV     #STBOT,SP       ;SET UP STACK POINTER
11497  050334  032777  000002  150722          BIT     #2,@SWR
11498  050342  001402                          BEQ     1$
11499  050344  000167  047714                  JMP     SLU1ST
11500  050350  012737  120162  000030  1$:     MOV     #ERROR4,@#30    ;SETUP FOR CORRECT ERROR CALL
11501  050356  012737  000003  001004          MOV     #3,@#$TESTN     ;PUT TEST NUMBER IN MAILBOX
11502
11503
11504                                  ;*************************************************************************
11505                                  ;TEST 420       LDFPS, STFPS AND DATA PATHS TEST
11506                                  ;*************************************************************************
11507  050364                  TS420:
11508  050364  012700  177777          MOV     #-1,R0                  ;INITIALIZE THE COUNT PATTERN.
11509  050370  012737  050442  000244  MOV     #AERR1,@#FPVECT         ;SET UP FOR UNABLE TO DECODE
11510  050376  012737  050442  000010  MOV     #AERR1,@#10             ;FPP INSTRUCTION TRAP TO 244 OR 10.
11511  050404  012737  050442  000004  MOV     #AERR1,@#ERRVECT        ;IF EITHER INSTRUCTION
11512                                                                  ;FAILS TO GO THROUGH THE
11513                                                                  ;CORRECT SRC OR DST MODE AN
11514                                                                  ;ODD ADDRESS TRAP WILL OCCUR.
11515  050412                  A1:
11516  050412  010004          A11:    MOV     R0,R4
11517  050414  042704  030020          BIC     #30020,R4
11518  050420  170104                  LDFPS   R4                      ;TEST INSTRUCTION.
11519
11520  050422  012701  177777          MOV     #-1,R1
11521  050426  170201          A12:    STFPS   R1                      ;TEST INSTRUCTION.
11522  050430  010004                  MOV     R0,R4                   ;MASK OFF UNSETTABLE BITS.
11523  050432  042704  030020          BIC     #30020,R4
11524  050436  020401                  CMP     R4,R1                   ;COMPARE DATA EXPECTED WITH
11525                                                                  ;THE DATA READ.
11526  050440  001401                  BEQ     A2
11527  050442                  AERR1:
11528  050442  104000                  EMT                     ;
11529
11530  050444  012700  000001  A2:     MOV     #1,R0                   ;NEXT PATTERN WILL BE ALL ZERO
11531  050450  077020                  SOB     R0,A1                   ;DECREMENT COUNT PATTERN
11532  050452                  ADONE:
11533  050452  004767  047522          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
11534                                                          ;SEE IF THE USER HAS EXPRESSED
11535                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
11536                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
11537                                                          ;THE USER TYPED CONTROL G?).
11538
11539
11540                                  ;*************************************************************************
11541                                  ;TEST 421       CFCC TEST
```

```
11542
11543  050456
11544  050456  012700  000017          ;*****************************************************************************
11545                                  TS421:
11546  050462                                  MOV     #17,R0                  ;R0 CONTAINS TO TEST PATTERN.
11547  050462  170100                  B1:
11548                                           LDFPS   R0                      ;LOAD THE TEST PATTERN
11549  050464
11550  050464  170000                  B2:
11551                                           CFCC                            ;COPY CONDITION CODES.
11552  050466  013703  177776
11553  050472  042703  177760                  MOV     @#PSW,R3                ;SEE IF PATTERN TRANSFERED.
11554  050476  020003                           BIC     #177760,R3
11555  050500  001401                           CMP     R0,R3
11556  050502  104000                           BEQ     B3
11557  050504  077012                           EMT                     ;
11558  050506                          B3:      SOB     R0,B1
11559  050506  004767  047466        BDONE:
11560                                           JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
11561                                                                   ;SEE IF THE USER HAS EXPRESSED
11562                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
11563                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
11564                                                                   ;THE USER TYPED CONTROL G?).
11565          ;*****************************************************************************
11566          ;TEST 422       SETF, SETD, SETI AND SETL TEST
11567          ;*****************************************************************************
11568  050512  TS422:
11569  050512  005000                   CLR     R0
11570
11571  050514  170100                   LDFPS   R0                      ;CLEAR THE FPS.
11572  050516  170001          C15:     SETF                            ;TEST INSTRUCTION.
11573
11574  050520  170201                   STFPS   R1                      ;GET RESULT.
11575  050522  005002                   CLR     R2
11576  050524  020201                   CMP     R2,R1                   ;DID AN ERROR OCCUR?
11577  050526  001401                   BEQ     C2
11578  050530  104000                   EMT                     ;
11579  050532  012700  147757  C2:      MOV     #147757,R0
11580
11581  050536  170100                   LDFPS   R0                      ;PUT 147757 IS FPS
11582  050540  170001          C25:     SETF                            ;CLEAR FD BIT.
11583
11584  050542  170201                   STFPS   R1                      ;GET RESULT
11585  050544  012702  147557           MOV     #147557,R2
11586  050550  020102                   CMP     R1,R2                   ;RESULT CORRECT.
11587  050552  001401                   BEQ     C3
11588  050554  104000                   EMT                     ;
11589  050556  012700  147757  C3:      MOV     #147757,R0
11590
11591  050562  170100                   LDFPS   R0                      ;LOAD 147757 INTO FPS.
11592  050564  170011          C35:     SETD                            ;SETD FD BIT.
11593
11594  050566  170201                   STFPS   R1
11595  050570  012702  147757           MOV     #147757,R2
11596  050574  020102                   CMP     R1,R2                   ;RESULT CORRECT?
11597  050576  001401                   BEQ     C4
```

```
11598  050600  104000                  EMT                     ;
11599  050602  005000          C4:     CLR     R0
11600  050604  170100                  LDFPS   R0                      ;CLEAR FPS.
11601  050606  170011          C45:    SETD                            ;SET FD BIT.
11602
11603  050610  170201                  STFPS   R1                      ;GET RESULT.
11604  050612  012702  000200        MOV     #200,R2
11605  050616  020102                  CMP     R1,R2                   ;RESULT CORRECT?
11606  050620  001401                  BEQ     C5
11607  050622  104000                  EMT                     ;
11608  050624  005000          C5:     CLR     R0
11609
11610  050626  170100                  LDFPS   R0                      ;CLEAR FPS
11611  050630  170002          C55:    SETI                            ;CLEAR FL BIT.
11612
11613  050632  170201                  STFPS   R1                      ;GET RESULT.
11614  050634  005002                  CLR     R2
11615  050636  020201                  CMP     R2,R1                   ;RESULT CORRECT?
11616  050640  001401                  BEQ     C6
11617  C50642  104000                  EMT                     ;
11618  050644  012700  147757  C6:     MOV     #147757,R0
11619  050650  170100                  LDFPS   R0                      ;PUT 147757 INTO FPS
11620  050652  170002          C65:    SETI                            ;CLEAR FL BIT.
11621
11622  050654  170201                  STFPS   R1                      ;GET THE RESULT.
11623  050656  012702  147657        MOV     #147657,R2
11624  050662  020102                  CMP     R1,R2                   ;RESULT CORRECT?
11625  050664  001401                  BEQ     C7
11626  050666  104000                  EMT                     ;
11627  050670  012700  147757  C7:     MOV     #147757,R0
11628  050674  170100                  LDFPS   R0                      ;SET FPS TO 147757.
11629  050676  170012          C75:    SETL                            ;SET FL BIT.
11630
11631  050700  170201                  STFPS   R1                      ;GET THE RESULT.
11632  050702  012702  147757        MOV     #147757,R2
11633  050706  020102                  CMP     R1,R2                   ;RESULT CORRECT?
11634  050710  001401                  BEQ     C8
11635  050712  104000                  EMT                     ;
11636  050714  005000          C8:     CLR     R0
11637  050716  170100                  LDFPS   R0                      ;CLEAR FPS.
11638  050720  170012          C85:    SETL                            ;SET FL BIT.
11639
11640  050722  170201                  STFPS   R1
11641  050724  012702  000100        MOV     #100,R2
11642  050730  020102                  CMP     R1,R2                   ;RESULT CORRECT.
11643  050732  001401                  BEQ     CDONE
11644  050734  104000                  EMT                     ;
11645  050736          CDONE:
11646  050736  004767  047236          JSR     PC,..RSET               ;GO INITIALIZE THE FPS AND STACK; AND
11647                                                          ;SEE IF THE USER HAS EXPRESSED
11648                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
11649                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
11650                                                          ;THE USER TYPED CONTROL G?).
11651
11652
11653                   ;*****************************************************************************
```

```
11654
11655
11656  050742
11657  050742  012705  170003          TS423:  MOV     #170003,R5              ;INITIAL OP CODE.
11658  050746  012737  051006  000004          MOV     #DERR2,@#ERRVECT
11659  050754  012737  051040  000244          MOV     #DERR1,@#FPVECT
11660
11661  050762  005000                  D1:     CLR     R0
11662  050764  170100                          LDFPS   R0                      ;CLEAR FPS.
11663  050766  005002                          CLR     R2
11664  050770  010537  050774                  MOV     R5,@#D2                 ;SET UP THE ILLEGAL INSTRUCTION.
11665  050774  000000                  D2:     .WORD   0
11666  050776  170000                  D3:     CFCC
11667  051000  005202                          INC     R2
11668  051002  005202                  D4:     INC     R2
11669
11670  051004  170201                          STFPS   R1                      ;REPORT FAILURE. DID NOT TRAP.
11671  051006                          DERR2:
11672  051006  104000                          EMT                     ;
11673  051010  022705  170010          D5:     CMP     #170010,R5              ;COMPUTE NEXT OP CODE
11674  051014  001003                          BNE     D6
11675  051016  012705  170013                  MOV     #170013,R5
11676  051022  000757                          BR      D1
11677
11678  051024  022705  170077          D6:     CMP     #170077,R5
11679  051030  001001                          BNE     D7
11680  051032  000424                          BR      DDONE
11681  051034  005205                  D7:     INC     R5
11682  051036  000751                          BR      D1
11683
11684  051040  022716  050776          DERR1:  CMP     #D3,(SP)                ;DID TRAP OCCUR ON TEST INSTRUCTION?
11685  051044  001401                          BEQ     1$
11686  051046  104000                          EMT                     ;
11687  051050  022626                  1$:     CMP     (SP)+,(SP)+
11688  051052  170201                          STFPS   R1                      ;GET THE FPS AND SEE IF IT IS
11689  051054  022701  100000                  CMP     #100000,R1              ;SET CORRECTLY.
11690  051060  001401                          BEQ     3$
11691  051062  104000                          EMT                     ;
11692  051064  012704  000001          3$:     MOV     #1,R4
11693  051070  170304                  D8:     STST    R4                      ;GET THE FEC CODE. NOTE THAT
11694                                                                          ;IF THE DESTINATION MODE IS
11695                                                                          ;IMPROPERLY DECODED AN ODD
11696                                                                          ;ADDRESS TRAP TO 4 SHOULD OCCUR.
11697  051072  022704  000002                  CMP     #2,R4                   ;WAS FEC CORRECT?
11698  051076  001001                          BNE     D9
11699  051100  000743                          BR      D5
11700
11701  051102                          D9:                                     ;REPORT STST FAILURE
11702  051102  104000                          EMT                     ;
11703  051104                          DDONE:
11704  051104  004767  047070                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
11705                                                                  ;SEE IF THE USER HAS EXPRESSED
11706                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
11707                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
11708                                                                  ;THE USER TYPED CONTROL G?).
11709
```

```
11710                                   ;*****************************************************************************
11711                                   ;TEST 424       FID, INTERRUPT DISABLE, BIT TEST
11712                                   ;*****************************************************************************
11713  051110                           TS424:
11714  051110  012737  051150  000244           MOV     #EERR0,@#FPVECT         ;SETUP FOR THE INTERRUPT.
11715
11716  051116  012700  040000          E1:      MOV     #40000,R0
11717  051122  170100                           LDFPS   R0                      ;SET FID.
11718  051124  170020                  E3:      .WORD   170020                  ;ILLEGAL FPP INSTRUCTION.
11719  051126  170000                  E4:      CFCC
11720
11721  051130  170201                           STFPS   R1                      ;SEE IF ERROR WAS DETECTED.
11722  051132  022701  140000                   CMP     #140000,R1
11723  051136  001004                           BNE     EERR0
11724
11725  051140  170304                           STST    R4                      ;SEE IF FEC=2
11726  051142  022704  000002                   CMP     #2,R4
11727  051146  001401                           BEQ     EDONE
11728  051150                          EERR0:
11729  051150  104000                           EMT                     ;
11730  051152                          EDONE:
11731  051152  004767  047022                   JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
11732                                                                   ;SEE IF THE USER HAS EXPRESSED
11733                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
11734                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
11735                                                                   ;THE USER TYPED CONTROL G?).
11736
11737
11738                                   ;*****************************************************************************
11739                                   ;TEST 425       LDD AND STD, WITH SRC AND DST MODE 1, TEST
11740                                   ;*****************************************************************************
11741  051156                           TS425:
11742
11743  051156  005000                           CLR     R0
11744  051160  170100                           LDFPS   R0
11745  051162  170011                           SETD
11746  051164  012701  051426                   MOV     #FDATI0,R1              ;SET UP THE LOAD DATA.
11747  051170  012702  051472                   MOV     #FXDAT0,R2
11748  051174  012703  000010                   MOV     #10,R3
11749
11750  051200  012221                  F2:      MOV     (R2)+,(R1)+
11751  051202  077302                           SOB     R3,F2
11752
11753  051204  012700  051436                   MOV     #FDATI4,R0              ;SETUP R0 FOR THE LDD (R0),AC0.
11754  051210  012737  051424  000004           MOV     #FERR20,@#ERRVECT       ;IF THE SRC FLOWS FAIL THEN
11755                                                                           ;AN ODD ADDRESS MAY OCCUR.
11756  051216  005003                           CLR     R3
11757
11758  051220  172410                  F3:      LDD     (R0),AC0
11759  051222  005203                  F4:      INC     R3
11760  051224  005203                           INC     R3
11761
11762  051226  020027  051436                   CMP     R0,#FDATI4              ;WAS R0 AFFECTED?
11763  051232  001401                           BEQ     F5
11764  051234  104000                           EMT                     ;
11765  051236  020327  000002          F5:      CMP     R3,#2                   ;SEE IF THE PC WAS ADVERSELY
```

```
11766  051242  001401                        BEQ     1$
11767  051244  104000                        EMT                     ;
11768  051246  012701  051426        1$:     MOV     #FDATI0,R1              ;MAKE SURE THE SOURCE DATA WAS
11769  051252  012702  051472                MOV     #FXDAT0,R2              ;NOT AFFECTED.
11770  051256  012703  000010                MOV     #10,R3
11771  051262  022122                2$:     CMP     (R1)+,(R2)+
11772  051264  001401                        BEQ     3$
11773  051266  104000                        EMT                     ;
11774  051270  077304                3$:     SOB     R3,2$
11775
11776  051272  170201                        STFPS   R1                      ;MAKE SURE THE FPS IS CORRECT.
11777  051274  022701  000200                CMP     #200,R1
11778  051300  001401                        BEQ     F6
11779  051302  104000                        EMT                     ;
11780  051304  012703  177777        F6:     MOV     #-1,R3
11781  051310  012704  000010                MOV     #10,R4
11782  051314  012705  051450                MOV     #FDAT00,R5              ;SET UP THE OUTPUT DATA BUFFER.
11783  051320  010325                F7:     MOV     R3,(R5)+
11784  051322  077402                        SOB     R4,F7
11785
11786  051324  012700  051460                MOV     #FDAT04,R0              ;SET UP R0 FOR DST MODE 1 REG 0.
11787  051330  012737  051424  000004        MOV     #FERR20,@#ERRVECT       ;IF THE DST FLOWS FAIL AN ODD
11788                                                                        ;ADDRESS COULD OCCUR.
11789  051336  005003                        CLR     R3
11790
11791  051340  174010                F10:    STD     AC0,(R0)                ;TEST INSTRUCTION.
11792  051342  005203                F11:    INC     R3
11793  051344  005203                        INC     R3
11794
11795  051346  020027  051460                CMP     R0,#FDAT04                      ;WAS R0 MODIFIED?
11796  051352  001401                        BEQ     F12
11797  051354  104000                        EMT                     ;
11798  051356  020327  000002        F12:    CMP     R3,#2                   ;WAS THE PC AFFECTED CORRECTLY?
11799  051362  001401                        BEQ     F135
11800  051364  104000                        EMT                     ;
11801  051366  012701  051450        F135:   MOV     #FDAT00,R1
11802  051372  012702  051472                MOV     #FXDAT0,R2
11803  051376  012703  000010                MOV     #10,R3          ;SETUP LOOP COUNT
11804  051402  022122                F13:    CMP     (R1)+,(R2)+     ;WAS DATA OUTPUT CORRECTLY
11805  051404  001401                        BEQ     F14
11806  051406  104000                        EMT                     ;
11807  051410  077304                F14:    SOB     R3,F13          ;SUBTRACT 1 FROM LOOP COUNT AND LOOP IF NOT ZERO
11808  051412  005001                F22:    CLR     R1
11809  051414  170201                        STFPS   R1                      ;MAKE SURE FPS IS CORRECT.
11810  051416  022701  000200                CMP     #200,R1
11811  051422  001433                        BEQ     FDONE
11812  051424                        FERR20:
11813  051424  104000                        EMT                     ;
11814
11815  051426  177777        FDATI0: -1
11816  051430  177777        FDATI1: -1
11817  051432  177777        FDATI2: -1
11818  051434  177777        FDATI3: -1
11819  051436  177777        FDATI4: -1
11820  051440  177777        FDATI5: -1
11821  051442  177777        FDATI6: -1
```

```
11822  051444  177777                FDATI7: -1
11823  051446  177777                        -1
11824  051450  177777                FDATO0: -1
11825  051452  177777                FDATO1: -1
11826  051454  177777                FDATO2: -1
11827  051456  177777                FDATO3: -1
11828  051460  177777                FDATO4: -1
11829  051462  177777                FDATO5: -1
11830  051464  177777                FDATO6: -1
11831  051466  177777                FDATO7: -1
11832  051470  177777                        -1
11833  051472  177777                FXDAT0: -1
11834  051474  177777                FXDAT1: -1
11835  051476  177777                FXDAT2: -1
11836  051500  177777                FXDAT3: -1
11837  051502  052525                FXDAT4: 052525
11838  051504  031463                FXDAT5: 031463
11839  051506  007417                FXDAT6: 007417
11840  051510  000477                FXDAT7: 000477
11841
11842
11843  051512                        FDONE:
11844  051512  004767  046462                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
11845                                                                ;SEE IF THE USER HAS EXPRESSED
11846                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
11847                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
11848                                                                ;THE USER TYPED CONTROL G?).
11849
11850
11851                                ;*****************************************************************************************
11852                                ;TEST 426       FSRC MODE 0 TEST
11853                                ;*****************************************************************************************
11854  051516                        TS426:
11855  051516                        I1:
11856  051516  170011                        SETD                    ;SET FD.
11857  051520  012700  052004                MOV     #IDATI0,R0
11858  051524  012701  051754                MOV     #IPATI0,R1
11859  051530  012702  000004                MOV     #4,R2
11860  051534  012120                I2:     MOV     (R1)+,(R0)+     ;SET UP THE INPUT DATA BUFFER.
11861  051536  077202                        SOB     R2,I2
11862
11863  051540  012700  052004                MOV     #IDATI0,R0      ;LOAD AC1
11864  051544  172510                        LDD     (R0),AC1
11865
11866  051546  012700  051764                MOV     #IPAT20,R0      ;LOAD AC0
11867  051552  172410                        LDD     (R0),AC0
11868
11869  051554  012701  000001                MOV     #1,R1           ;IN CASE THE FSRC FLOWS FAIL
11870  051560  172401                I3:     LDD     AC1,AC0         ;TEST INSTRUCTION.
11871  051562  000240                I4:     NOP
11872  051564  000240                I5:     NOP
11873
11874  051566  012700  051774                MOV     #IDATO0,R0
11875  051572  174010                        STD     AC0,(R0)        ;GET AC0, THE RESULTS.
11876
11877  051574  012700  051774                MOV     #IDATO0,R0      ;SEE IF DATA IS CORRECT.
```

```
11878  051600  012701  052004          MOV     #IDATI0,R1
11879  051604  012702  000004          MOV     #4,R2
11880  051610  022021          I6:     CMP     (R0)+,(R1)+
11881  051612  001401                  BEQ     I105
11882  051614  104000                  EMT                     ;
11883  051616  077204          I105:   SOB     R2,I6
11884
11885                          ;NOW TEST THE LOAD INSTRUCTION WITH FSRC MODE ZERO AND FD CLEAR.
11886
11887  051620  012700  051754  I12:    MOV     #IPAT10,R0
11888  051624  012701  052004          MOV     #IDATI0,R1
11889  051630  012702  000004          MOV     #4,R2
11890  051634  012021          I13:    MOV     (R0)+,(R1)+
11891  051636  077202                  SOB     R2,I13
11892
11893  051640  012700  052004          MOV     #IDATI0,R0              ;SET UP AC1
11894  051644  172510                  LDD     (R0),AC1
11895
11896  051646  012700  051764          MOV     #IPAT20,R0              ;SET UP AC0
11897  051652  172410                  LDD     (R0),AC0
11898
11899  051654  012701  000001          MOV     #1,R1
11900  051660  170001                  SETF                            ;CLEAR FD.
11901
11902  051662  172401          I14:    LDF     AC1,AC0                 ;TEST INSTRUCTION.
11903  051664  000240          I15:    NOP
11904  051666  000240          I16:    NOP
11905
11906  051670  170200                  STFPS   R0                      ;SEE IF FPS IS STILL CLEAR.
11907  051672  022700  000004          CMP     #4,R0
11908  051676  001401                  BEQ     I17
11909  051700  104000                  EMT                     ;
11910  051702                  I17:                            ;RESET TO DOUBLE MODE.
11911  051702  170011                  SETD
11912
11913  051704  012700  051774          MOV     #IDATO0,R0
11914  051710  174010                  STD     AC0,(R0)                ;GET AC0
11915
11916  051712  012737  177777  052010  MOV     #-1,@#IDATI2
11917  051720  012737  177777  052012  MOV     #-1,@#IDATI3
11918  051726  012700  051774          MOV     #IDATO0,R0
11919  051732  012701  052004          MOV     #IDATI0,R1
11920  051736  012702  000004          MOV     #4,R2
11921  051742  022021          I20:    CMP     (R0)+,(R1)+             ;SEE IF AC0 WAS CORRECT.
11922  051744  001401                  BEQ     I23
11923  051746  104000                  EMT                     ;
11924  051750  077204          I23:    SOB     R2,I20
11925  051752  000420                  BR      IDONE                   ;NO ERRORS.
11926
11927  051754  000000          IPAT10: 0
11928  051756  170360          IPAT11: 170360
11929  051760  016161          IPAT12: 016161
11930  051762  052525          IPAT13: 052525
11931
11932  051764  177777          IPAT20: -1
11933  051766  177777          IPAT21: -1
```

```
11934  051770  177777                  IPAT22: -1
11935  051772  177777                  IPAT23: -1
11936
11937  051774  000000                  IDAT00: 0
11938  051776  000000                  IDAT01: 0
11939  052000  000000                  IDAT02: 0
11940  052002  000000                  IDAT03: 0
11941
11942  052004  000000                  IDATI0: 0
11943  052006  000000                  IDATI1: 0
11944  052010  000000                  IDATI2: 0
11945  052012  000000                  IDATI3: 0
11946
11947  052014                          IDONE:
11948  052014  004767  046160                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
11949                                                                  ;SEE IF THE USER HAS EXPRESSED
11950                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
11951                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
11952                                                                  ;THE USER TYPED CONTROL G?).
11953
11954
11955                                  ;*****************************************************************************
11956                                  ;TEST 427       FDST MODE 0 TEST
11957                                  ;*****************************************************************************
11958  052020                          TS427:
11959  052020  170011                          SETD                            ;SET FD
11960  052022  012700  052260                  MOV     #TPAT10,R0
11961  052026  012701  052310                  MOV     #TDAT10,R1
```

```
                                                                M 2
 11962  052032  012702  000004                MOV     #4,R2
```

```
11963  052036  012021
11964  052040  077202
11965

                        T2:     MOV     (R0)+,(R1)+             ;SET UP THE INPUT DATA BUFFER.
                                SOB     R2,T2
```

```
T427    FDST MODE 0 TEST

11966  052042  012700  052310          MOV     #TDATI0,R0              ;LOAD AC0
11967  052046  172410                  LDD     (R0),AC0
11968
11969  052050  012700  052270          MOV     #TPAT20,R0              ;LOAD AC1
11970  052054  172510                  LDD     (R0),AC1
11971
11972  052056  012701  000001          MOV     #1,R1                   ;IF THE (BUT FDST) FORK FAILS
11973  052062  174001          T3:     STD     AC0,AC1
11974  052064  000240          T4:     NOP
11975  052066  000240          T5:     NOP
11976
11977  052070  012700  052300          MOV     #TDAT00,R0
11978  052074  174110                  STD     AC1,(R0)                ;GET THE DATA.
11979
11980  052076  012703  052300          MOV     #TDAT00,R3              ;SEE IF THE DATA IS CORRECT.
11981  052102  012704  052310          MOV     #TDATI0,R4
11982  052106  012705  000004          MOV     #4,R5
11983  052112  022324          T6:     CMP     (R3)+,(R4)+
11984  052114  001401                  BEQ     T105
11985  052116  104000                  EMT                             ;
11986  052120  077504          T105:   SOB     R5,T6
11987
11988                           ;NOW TEST THE STF AC0,AC1 INSTRUCTION.
11989
11990  052122  012700  052260  T12:    MOV     #TPAT10,R0              ;SET UP THE INPUT DATA BUFFER.
11991  052126  012701  052310          MOV     #TDATI0,R1
11992  052132  012702  000004          MOV     #4,R2
11993  052136  012021          T13:    MOV     (R0)+,(R1)+
11994  052140  077202                  SOB     R2,T13
11995
11996  052142  012700  052310          MOV     #TDATI0,R0              ;SET UP AC0
11997  052146  172410                  LDD     (R0),AC0
11998
11999  052150  012700  052270          MOV     #TPAT20,R0              ;SET UP AC1
12000  052154  172510                  LDD     (R0),AC1
12001
12002  052156  012701  000001          MOV     #1,R1
12003  052162  170001                  SETF                            ;CLEAR FD
12004  052164  174001          T14:    STF     AC0,AC1
12005  052166  000240          T15:    NOP
12006  052170  000240          T16:    NOP
12007
12008  052172  005000                  CLR     R0
12009  052174  170200                  STFPS   R0                      ;SEE IF FPS IS CLEAR.
12010  052176  022700  000010          CMP     #10,R0
12011  052202  001401                  BEQ     T17
12012  052204  104000                  EMT                             ;
12013  052206          T17:
12014  052206  170011                  SETD                            ;SET FD.
12015
12016  052210  012700  052300          MOV     #TDAT00,R0
12017  052214  174110                  STD     AC1,(R0)                ;PICK UP AC1.
12018
12019  052216  012737  177777  052314  MOV     #-1,@#TDATI2
12020  052224  012737  177777  052316  MOV     #-1,@#TDATI3
12021  052232  012703  052300          MOV     #TDAT00,R3
```

```
12022  052236  012704  052310          MOV     #TDATI0,R4
12023  052242  012705  000004          MOV     #4,R5
12024  052246  022324          T20:    CMP     (R3)+,(R4)+             ;WAS THE DATA TRANSFERRED CORRECTLY?
12025  052250  001401                  BEQ     T23
12026  052252  104000                  EMT                             ;
12027  052254  077504          T23:    SOB     R5,T20
12028  052256  000420                  BR      TDONE
12029
12030
12031  052260  000000          TPAT10: 0
12032  052262  170360          TPAT11: 170360
12033  052264  016161          TPAT12: 016161
12034  052266  052525          TPAT13: 052525
12035
12036  052270  177777          TPAT20: -1
12037  052272  177777          TPAT21: -1
12038  052274  177777          TPAT22: -1
12039  052276  177777          TPAT23: -1
12040
12041  052300  000000          TDAT00: 0
12042  052302  000000          TDAT01: 0
12043  052304  000000          TDAT02: 0
12044  052306  000000          TDAT03: 0
12045
12046  052310  000000          TDATI0: 0
12047  052312  000000          TDATI1: 0
12048  052314  000000          TDATI2: 0
12049  052316  000000          TDATI3: 0
12050
12051  052320                  TDONE:
12052  052320  004767  045654          JSR     PC,..RSET               ;GO INITIALIZE THE FPS AND STACK; AND
12053                                                                  ;SEE IF THE USER HAS EXPRESSED
12054                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
12055                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12056                                                                  ;THE USER TYPED CONTROL G?).
12057
12058
12059
12060                           ;*************************************************************************************
12061                           ;TEST 430       ACCUMULATORS DATA PATTERNS TEST
12062                           ;*************************************************************************************
12063  052324                  TS430:
12064  052324  170011                  SETD                            ;SET FD.
12065                          ;TEST ACCUMULATOR 0 WITH FLOATING ONE
12066  052326  012700  054060          MOV     #GPAT00,R0
12067  052332  012701  054120          MOV     #GDAT00,R1
12068  052336  004737  053732          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12069  052342  012703  000102          MOV     #102,R3
12070  052346                  G1:
12071  052346  172410                  LDD     (R0),AC0
12072  052350  174000                  STD     AC0,AC0
12073  052352  172400                  LDD     AC0,AC0                 ;STORE THE TEST PATTERN.
12074  052354  174011                  STD     AC0,(R1)
12075  052356  004737  054030          JSR     PC,@#GCMP                       ;COMPARE THE DATA READ WITH
12076                                                                  ;THAT WHICH WAS WRITTEN.
12077  052362  005737  054054          TST     @#GFLAG1
```

```
12078  052366  001004
12079  052370  005137  054054          BNE     G2
12080  052374  000261                  COM     @#GFLAG1
12081  052376  000401                  SEC
12082  052400  000241                  BR      G3
12083  052402  006160  000006  G2:     CLC
12084  052406  006160  000004  G3:     ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12085  052412  006160  000002          ROL     4(R0)
12086  052416  006110                  ROL     2(R0)
12087  052420  004737  054010          ROL     (R0)
12088                                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                                                       ;BUFFER.
12089  052424  077330                  SOB     R3,G1
12090
12091                          ;TEST ACCUMULATOR 0 WITH FLOATING ZERO
12092  052426  012700  054070          MOV     #GPAT10,R0
12093  052432  012701  054120          MOV     #GDAT00,R1
12094  052436  004737  053732          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12095  052442  012703  000102          MOV     #102,R3
12096  052446                  G4:
12097  052446  172410                  LDD     (R0),AC0
12098  052450  174000                  STD     AC0,AC0
12099  052452  172400                  LDD     AC0,AC0         ;STORE THE TEST PATTERN.
12100  052454  174011                  STD     AC0,(R1)
12101  052456  004737  054030          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12102                                                                 ;THAT WHICH WAS WRITTEN.
12103  052462  005737  054054          TST     @#GFLAG1
12104  052466  001004                  BNE     G5
12105  052470  005137  054054          COM     @#GFLAG1
12106  052474  000241                  CLC
12107  052476  000401                  BR      G6
12108  052500  000261          G5:     SEC
12109  052502  006160  000006  G6:     ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12110  052506  006160  000004          ROL     4(R0)
12111  052512  006160  000002          ROL     2(R0)
12112  052516  006110                  ROL     (R0)
12113  052520  004737  054010          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12114                                                                 ;BUFFER.
12115  052524  077330                  SOB     R3,G4
12116
12117                          ;TEST ACCUMULATOR 1 WITH FLOATING ONE
12118  052526  012700  054060          MOV     #GPAT00,R0
12119  052532  012701  054120          MOV     #GDAT00,R1
12120  052536  004737  053732          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12121  052542  012703  000102          MOV     #102,R3
12122  052546                  G7:
12123  052546  172410                  LDD     (R0),AC0
12124  052550  174001                  STD     AC0,AC1
12125  052552  172401                  LDD     AC1,AC0         ;STORE THE TEST PATTERN.
12126  052554  174011                  STD     AC0,(R1)
12127  052556  004737  054030          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12128                                                                 ;THAT WHICH WAS WRITTEN.
12129  052562  005737  054054          TST     @#GFLAG1
12130  052566  001004                  BNE     G10
12131  052570  005137  054054          COM     @#GFLAG1
12132  052574  000261                  SEC
12133  052576  000401                  BR      G11
```

```
12134  052600  000241
12135  052602  006160  000006
12136  052606  006160  000004
12137  052612  006160  000002
12138  052616  006110
12139  052620  004737  054010
12140
12141  052624  077330
12142
12143
12144  052626  012700  054070
12145  052632  012701  054120
12146  052636  004737  053732
12147  052642  012703  000102
12148  052646
12149  052646  172410
12150  052650  174001
12151  052652  172401
12152  052654  174011
12153  052656  004737  054030
12154
12155  052662  005737  054054
12156  052666  001004
12157  052670  005137  054054
12158  052674  000241
12159  052676  000401
12160  052700  000261
12161  052702  006160  000006
12162  052706  006160  000004
12163  052712  006160  000002
12164  052716  006110
12165  052720  004737  054010
12166
12167  052724  077330
12168
12169
12170  052726  012700  054060
12171  052732  012701  054120
12172  052736  004737  053732
12173  052742  012703  000102
12174  052746
12175  052746  172410
12176  052750  174002
12177  052752  172402
12178  052754  174011
12179  052756  004737  054030
12180
12181  052762  005737  054054
12182  052766  001004
12183  052770  005137  054054
12184  052774  000261
12185  052776  000401
12186  053000  000241
12187  053002  006160  000006
12188  053006  006160  000004
12189  053012  006160  000002
```

```
G10:    CLC
G11:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
        ROL     4(R0)
        ROL     2(R0)
        ROL     (R0)
        JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                        ;BUFFER.
        SOB     R3,G7

;TEST ACCUMULATOR 1 WITH FLOATING ZERO
        MOV     #GPAT10,R0
        MOV     #GDAT00,R1
        JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
        MOV     #102,R3
G12:
        LDD     (R0),AC0
        STD     AC0,AC1
        LDD     AC1,AC0         ;STORE THE TEST PATTERN.
        STD     AC0,(R1)
        JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                        ;THAT WHICH WAS WRITTEN.
        TST     @#GFLAG1
        BNE     G13
        COM     @#GFLAG1
        CLC
        BR      G14
G13:    SEC
G14:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
        ROL     4(R0)
        ROL     2(R0)
        ROL     (R0)
        JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
                                        ;BUFFER.
        SOB     R3,G12

;TEST ACCUMULATOR 2 WITH FLOATING ONE
        MOV     #GPAT00,R0
        MOV     #GDAT00,R1
        JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
        MOV     #102,R3
G15:
        LDD     (R0),AC0
        STD     AC0,AC2
        LDD     AC2,AC0         ;STORE THE TEST PATTERN.
        STD     AC0,(R1)
        JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
                                        ;THAT WHICH WAS WRITTEN.
        TST     @#GFLAG1
        BNE     G16
        COM     @#GFLAG1
        SEC
        BR      G17
G16:    CLC
G17:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
        ROL     4(R0)
        ROL     2(R0)
```

```
12190  053016  006110                          ROL     (R0)
12191  053020  004737  054010                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12192                                                                          ;BUFFER.
12193  053024  077330                          SOB     R3,G15
12194
12195                                  ;TEST ACCUMULATOR 2 WITH FLOATING ZERO
12196  053026  012700  054070                  MOV     #GPAT10,R0
12197  053032  012701  054120                  MOV     #GDAT00,R1
12198  053036  004737  053732                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12199  053042  012703  000102                  MOV     #102,R3
12200  053046                          G20:
12201  053046  172410                          LDD     (R0),AC0
12202  053050  174002                          STD     AC0,AC2
12203  053052  172402                          LDD     AC2,AC0         ;STORE THE TEST PATTERN.
12204  053054  174011                          STD     AC0,(R1)
12205  053056  004737  054030                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12206                                                                          ;THAT WHICH WAS WRITTEN.
12207  053062  005737  054054                  TST     @#GFLAG1
12208  053066  001004                          BNE     G21
12209  053070  005137  054054                  COM     @#GFLAG1
12210  053074  000241                          CLC
12211  053076  000401                          BR      G22
12212  053100  000261                  G21:    SEC
12213  053102  006160  000006          G22:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12214  053106  006160  000004                  ROL     4(R0)
12215  053112  006160  000002                  ROL     2(R0)
12216  053116  006110                          ROL     (R0)
12217  053120  004737  054010                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12218                                                                          ;BUFFER.
12219  053124  077330                          SOB     R3,G20
12220
12221                                  ;TEST ACCUMULATOR 3 WITH FLOATING ONE
12222  053126  012700  054060                  MOV     #GPAT00,R0
12223  053132  012701  054120                  MOV     #GDAT00,R1
12224  053136  004737  053732                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12225  053142  012703  000102                  MOV     #102,R3
12226  053146                          G23:
12227  053146  172410                          LDD     (R0),AC0
12228  053150  174003                          STD     AC0,AC3
12229  053152  172403                          LDD     AC3,AC0         ;STORE THE TEST PATTERN.
12230  053154  174011                          STD     AC0,(R1)
12231  053156  004737  054030                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12232                                                                          ;THAT WHICH WAS WRITTEN.
12233  053162  005737  054054                  TST     @#GFLAG1
12234  053166  001004                          BNE     G24
12235  053170  005137  054054                  COM     @#GFLAG1
12236  053174  000261                          SEC
12237  053176  000401                          BR      G25
12238  053200  000241                  G24:    CLC
12239  053202  006160  000006          G25:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12240  053206  006160  000004                  ROL     4(R0)
12241  053212  006160  000002                  ROL     2(R0)
12242  053216  006110                          ROL     (R0)
12243  053220  004737  054010                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12244                                                                          ;BUFFER.
12245  053224  077330                          SOB     R3,G23
```

```
12246
12247
12248  053226  012700  054070          ;TEST ACCUMULATOR 3 WITH FLOATING ZERO
12249  053232  012701  054120                  MOV     #GPAT10,R0
12250  053236  004737  053732                  MOV     #GDAT00,R1
12251  053242  012703  000102                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12252  053246                                  MOV     #102,R3
12253  053246  172410          G26:            LDD     (R0),AC0
12254  053250  174003                          STD     AC0,AC3
12255  053252  172403                          LDD     AC3,AC0         ;STORE THE TEST PATTERN.
12256  053254  174011                          STD     AC0,(R1)
12257  053256  004737  054030                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12258                                                                          ;THAT WHICH WAS WRITTEN.
12259  053262  005737  054054                  TST     @#GFLAG1
12260  053266  001004                          BNE     G27
12261  053270  005137  054054                  COM     @#GFLAG1
12262  053274  000241                          CLC
12263  053276  000401                          BR      G30
12264  053300  000261          G27:            SEC
12265  053302  006160  000006  G30:            ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12266  053306  006160  000004                  ROL     4(R0)
12267  053312  006160  000002                  ROL     2(R0)
12268  053316  006110                          ROL     (R0)
12269  053320  004737  054010                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12270                                                                          ;BUFFER.
12271  053324  077330                          SOB     R3,G26
12272
12273                                  ;TEST ACCUMULATOR 4 WITH FLOATING ONE
12274  053326  012700  054060                  MOV     #GPAT00,R0
12275  053332  012701  054120                  MOV     #GDAT00,R1
12276  053336  004737  053732                  JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12277  053342  012703  000102                  MOV     #102,R3
12278  053346          G31:
12279  053346  172410                          LDD     (R0),AC0
12280  053350  174004                          STD     AC0,AC4
12281  053352  172404                          LDD     AC4,AC0         ;STORE THE TEST PATTERN.
12282  053354  174011                          STD     AC0,(R1)
12283  053356  004737  054030                  JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12284                                                                          ;THAT WHICH WAS WRITTEN.
12285  053362  005737  054054                  TST     @#GFLAG1
12286  053366  001004                          BNE     G32
12287  053370  005137  054054                  COM     @#GFLAG1
12288  053374  000261                          SEC
12289  053376  000401                          BR      G33
12290  053400  000241          G32:            CLC
12291  053402  006160  000006  G33:            ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12292  053406  006160  000004                  ROL     4(R0)
12293  053412  006160  000002                  ROL     2(R0)
12294  053416  006110                          ROL     (R0)
12295  053420  004737  054010                  JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12296                                                                          ;BUFFER.
12297  053424  077330                          SOB     R3,G31
12298
12299                                  ;TEST ACCUMULATOR 4 WITH FLOATING ZERO
12300  053426  012700  054070                  MOV     #GPAT10,R0
12301  053432  012701  054120                  MOV     #GDAT00,R1
```

```
12302  053436  004737  053732          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12303  053442  012703  000102          MOV     #102,R3
12304  053446                  G34:
12305  053446  172410                  LDD     (R0),AC0
12306  053450  174004                  STD     AC0,AC4
12307  053452  172404                  LDD     AC4,AC0         ;STORE THE TEST PATTERN.
12308  053454  174011                  STD     AC0,(R1)
12309  053456  004737  054030          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12310                                                                  ;THAT WHICH WAS WRITTEN.
12311  053462  005737  054054          TST     @#GFLAG1
12312  053466  001004                  BNE     G35
12313  053470  005137  054054          COM     @#GFLAG1
12314  053474  000241                  CLC
12315  053476  000401                  BR      G36
12316  053500  000261          G35:    SEC
12317  053502  006160  000006  G36:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12318  053506  006160  000004          ROL     4(R0)
12319  053512  006160  000002          ROL     2(R0)
12320  053516  006110                  ROL     (R0)
12321  053520  004737  054010          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12322                                                                  ;BUFFER.
12323  053524  077330                  SOB     R3,G34
12324
12325                  ;TEST ACCUMULATOR 5 WITH FLOATING ONE
12326  053526  012700  054060          MOV     #GPAT00,R0
12327  053532  012701  054120          MOV     #GDAT00,R1
12328  053536  004737  053732          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12329  053542  012703  000102          MOV     #102,R3
12330  053546                  G37:
12331  053546  172410                  LDD     (R0),AC0
12332  053550  174005                  STD     AC0,AC5
12333  053552  172405                  LDD     AC5,AC0         ;STORE THE TEST PATTERN.
12334  053554  174011                  STD     AC0,(R1)
12335  053556  004737  054030          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12336                                                                  ;THAT WHICH WAS WRITTEN.
12337  053562  005737  054054          TST     @#GFLAG1
12338  053566  001004                  BNE     G40
12339  053570  005137  054054          COM     @#GFLAG1
12340  053574  000261                  SEC
12341  053576  000401                  BR      G41
12342  053600  000241          G40:    CLC
12343  053602  006160  000006  G41:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12344  053606  006160  000004          ROL     4(R0)
12345  053612  006160  000002          ROL     2(R0)
12346  053616  006110                  ROL     (R0)
12347  053620  004737  054010          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12348                                                                  ;BUFFER.
12349  053624  077330                  SOB     R3,G37
12350
12351                  ;TEST ACCUMULATOR 5 WITH FLOATING ZERO
12352  053626  012700  054070          MOV     #GPAT10,R0
12353  053632  012701  054120          MOV     #GDAT00,R1
12354  053636  004737  053732          JSR     PC,@#GSETUP             ;LOAD TEST PATTERN.
12355  053642  012703  000102          MOV     #102,R3
12356  053646                  G42:
12357  053646  172410                  LDD     (R0),AC0
```

```
12358  053650  174005                  STD     AC0,AC5
12359  053652  172405                  LDD     AC5,AC0         ;STORE THE TEST PATTERN.
12360  053654  174011                  STD     AC0,(R1)
12361  053656  004737  054030          JSR     PC,@#GCMP               ;COMPARE THE DATA READ WITH
12362                                                                   ;THAT WHICH WAS WRITTEN.
12363  053662  005737  054054          TST     @#GFLAG1
12364  053666  001004                  BNE     G43
12365  053670  005137  054054          COM     @#GFLAG1
12366  053674  000241                  CLC
12367  053676  000401                  BR      G44
12368  053700  000261          G43:    SEC
12369  053702  006160  000006  G44:    ROL     6(R0)                   ;GENERATE THE NEXT TEST PATTERN.
12370  053706  006160  000004          ROL     4(R0)
12371  053712  006160  000002          ROL     2(R0)
12372  053716  006110                  ROL     (R0)
12373  053720  004737  054010          JSR     PC,@#GRESET             ;RESET DEFAULT PATTERN IN OUTPUT
12374                                                                   ;BUFFER.
12375  053724  077330                  SOB     R3,G42
12376
12377
12378  053726  000137  054130          JMP     @#GDONE
12379
12380                          ;USE THIS ROUTINE TO INITIALIZE ALL THE DATA BUFFERS.
12381  053732  012705  054054  GSETUP: MOV     #GFLAG1,R5
12382  053736  012704  000026          MOV     #26,R4
12383  053742  005025          1$:     CLR     (R5)+
12384  053744  077402                  SOB     R4,1$
12385
12386  053746  012705  054070          MOV     #GPAT10,R5
12387  053752  012704  000010          MOV     #10,R4
12388  053756  005125          2$:     COM     (R5)+
12389  053760  077402                  SOB     R4,2$
12390
12391  053762  020067  000072  GS1:    CMP     R0,GPAT00
12392  053766  001401                  BEQ     3$
12393  053770  000207                  RTS     PC
12394
12395  053772  012705  054120  3$:     MOV     #GDAT00,R5
12396  053776  012704  000004          MOV     #4,R4
12397  054002  005125          4$:     COM     (R5)+
12398  054004  077402                  SOB     R4,4$
12399  054006  000207                  RTS     PC
12400
12401  054010  012705  054120  GRESET: MOV     #GDAT00,R5
12402  054014  012704  000004          MOV     #4,R4
12403  054020  005025          1$:     CLR     (R5)+
12404  054022  077402                  SOB     R4,1$
12405  054024  000137  053762          JMP     @#GS1
12406
12407                          ;SEE IF THE DATA WRITTEN MATCHES THE DATA READ.
12408  054030  012705  054120  GCMP:   MOV     #GDAT00,R5
12409  054034  012704  000004          MOV     #4,R4
12410  054040  010002                  MOV     R0,R2
12411  054042  022225          1$:     CMP     (R2)+,(R5)+
12412  054044  001401                  BEQ     2$
12413  054046  104000                  EMT                             ;
```

```
12414  054050  077404          2$:     SOB     R4,1$
12415  054052  000207                  RTS     PC
12416
12417
12418
12419
12420  054054  000000          GFLAG1: 0
12421  054056  000000          GFLAG2: 0
12422
12423  054060  000000          GPAT00: 0
12424  054062  000000          GPAT01: 0
12425  054064  000000          GPAT02: 0
12426  054066  000000          GPAT03: 0
12427
12428  054070  177777          GPAT10: -1
12429  054072  177777          GPAT11: -1
12430  054074  177777          GPAT12: -1
12431  054076  177777          GPAT13: -1
12432
12433  054100  177777          GAND0:  -1
12434  054102  177777          GAND1:  -1
12435  054104  177777          GAND2:  -1
12436  054106  177777          GAND3:  -1
12437
12438  054110  000000          GOR0:   0
12439  054112  000000          GOR1:   0
12440  054114  000000          GOR2:   0
12441  054116  000000          GOR3:   0
12442
12443  054120  000000          GDAT00: 0
12444  054122  000000          GDAT01: 0
12445  054124  000000          GDAT02: 0
12446  054126  000000          GDAT03: 0
12447
12448
12449  054130                  GDONE:
12450  054130  004767  044044          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
12451                                                          ;SEE IF THE USER HAS EXPRESSED
12452                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
12453                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12454                                                          ;THE USER TYPED CONTROL G?).
12455
12456
12457                          ;*****************************************************************************
12458                          ;TEST 431       FPP ACCUMULATORS DUAL ADDRESS TEST
12459                          ;*****************************************************************************
12460  054134                  TS431:
12461  054134  005037  054624  H1:     CLR     @#HFLAG
12462  054140  012700  054626          MOV     #HA1W,R0                ;INITIALIZE THE LOAD BUFFER DATA.
12463  054144  012701  054746          MOV     #HDAT1,R1
12464  054150  012703  000024          MOV     #24,R3
12465  054154  012120          H2:     MOV     (R1)+,(R0)+
12466  054156  077302                  SOB     R3,H2
12467
12468  054160  004767  000420          JSR     PC,HCLR         ;CLEAR THE OUTPUT DATA BUFFER.
12469
```

```
12470  054164  170011                H3:     SETD
12471                                ;LOAD ACCUMULATOR 1
12472  054166  012700  054626               MOV     #HA1W,R0
12473  054172  172410                       LDD     (R0),AC0
12474  054174  174001                       STD     AC0,AC1
12475                                ;LOAD ACCUMULATOR 2
12476  054176  012700  054636               MOV     #HA2W,R0
12477  054202  172410                       LDD     (R0),AC0
12478  054204  174002                       STD     AC0,AC2
12479                                ;LOAD ACCUMULATOR 3
12480  054206  012700  054646               MOV     #HA3W,R0
12481  054212  172410                       LDD     (R0),AC0
12482  054214  174003                       STD     AC0,AC3
12483                                ;LOAD ACCUMULATOR 4
12484  054216  012700  054656               MOV     #HA4W,R0
12485  054222  172410                       LDD     (R0),AC0
12486  054224  174004                       STD     AC0,AC4
12487                                ;LOAD ACCUMULATOR 5
12488  054226  012700  054666               MOV     #HA5W,R0
12489  054232  172410                       LDD     (R0),AC0
12490  054234  174005                       STD     AC0,AC5
12491
12492  054236  004737  054472       H4:     JSR     PC,@#HSTD       ;GO READ ALL ACCUMULATORS BACK.
12493
12494  054242  004737  054550               JSR     PC,@#HCMP       ;SEE IF DATA IS CORRECT.
12495
12496                                ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 1,
12497                                ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
12498                                ;THE DATA.
12499  054246  012700  054626               MOV     #HA1W,R0
12500  054252  012702  000004               MOV     #4,R2
12501  054256  010001                       MOV     R0,R1
12502  054260  005121               H5:     COM     (R1)+
12503  054262  172410                       LDD     (R0),AC0
12504  054264  174001                       STD     AC0,AC1
12505  054266  004737  054472               JSR     PC,@#HSTD       ;READ ALL THE ACCUMULATORS BACK.
12506  054272  004737  054550               JSR     PC,@#HCMP       ;CHECK THE DATA.
12507  054276  077210                       SOB     R2,H5
12508
12509                                ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 2,
12510                                ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
12511                                ;THE DATA.
12512  054300  012700  054636               MOV     #HA2W,R0
12513  054304  012702  000004               MOV     #4,R2
12514  054310  010001                       MOV     R0,R1
12515  054312  005121               H6:     COM     (R1)+
12516  054314  172410                       LDD     (R0),AC0
12517  054316  174002                       STD     AC0,AC2
12518  054320  004737  054472               JSR     PC,@#HSTD       ;READ ALL THE ACCUMULATORS BACK.
12519  054324  004737  054550               JSR     PC,@#HCMP       ;CHECK THE DATA.
12520  054330  077210                       SOB     R2,H6
12521
12522                                ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 3,
12523                                ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
12524                                ;THE DATA.
12525  054332  012700  054646               MOV     #HA3W,R0
```

```
12526  054336  012702  000004          MOV     #4,R2
12527  054342  010001                  MOV     R0,R1
12528  054344  005121          H7:     COM     (R1)+
12529  054346  172410                  LDD     (R0),AC0
12530  054350  174003                  STD     AC0,AC3
12531  054352  004737  054472          JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
12532  054356  004737  054550          JSR     PC,@#HCMP               ;CHECK THE DATA.
12533  054362  077210                  SOB     R2,H7
12534
12535                                  ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 4,
12536                                  ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
12537                                  ;THE DATA.
12538  054364  012700  054656          MOV     #HA4W,R0
12539  054370  012702  000004          MOV     #4,R2
12540  054374  010001                  MOV     R0,R1
12541  054376  005121          H10:    COM     (R1)+
12542  054400  172410                  LDD     (R0),AC0
12543  054402  174004                  STD     AC0,AC4
12544  054404  004737  054472          JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
12545  054410  004737  054550          JSR     PC,@#HCMP               ;CHECK THE DATA.
12546  054414  077210                  SOB     R2,H10
12547
12548                                  ;COMPLIMENT EACH WORD OF THE DATA STORED IN ACCUMULATOR 5,
12549                                  ;RELOAD THAT ACCUMULATOR, READ ALL THE ACCUMULATORS BACK AND CHECK
12550                                  ;THE DATA.
12551  054416  012700  054666          MOV     #HA5W,R0
12552  054422  012702  000004          MOV     #4,R2
12553  054426  010001                  MOV     R0,R1
12554  054430  005121          H11:    COM     (R1)+
12555  054432  172410                  LDD     (R0),AC0
12556  054434  174005                  STD     AC0,AC5
12557  054436  004737  054472          JSR     PC,@#HSTD               ;READ ALL THE ACCUMULATORS BACK.
12558  054442  004737  054550          JSR     PC,@#HCMP               ;CHECK THE DATA.
12559  054446  077210                  SOB     R2,H11
12560
12561
12562  054450  005737  054624          TST     @#HFLAG
12563  054454  001402                  BEQ     H12
12564  054456  000137  055016          JMP     @#HDONE
12565
12566  054462  005137  054624  H12:    COM     @#HFLAG
12567  054466  000137  054164          JMP     @#H3
12568
12569                                  ;STORE ALL ACCUMULATORS IN THE OUTPUT BUFFERS.
12570  054472  004737  054604  HSTD:   JSR     PC,@#HCLR               ;CLEAR ALL OUTPUT BUFFERS.
12571                                  ;STORE ACCUMULATOR 1
12572  054476  012704  054676          MOV     #HA1R,R4
12573  054502  172401                  LDD     AC1,AC0
12574  054504  174014                  STD     AC0,(R4)
12575                                  ;STORE ACCUMULATOR 2
12576  054506  012704  054706          MOV     #HA2R,R4
12577  054512  172402                  LDD     AC2,AC0
12578  054514  174014                  STD     AC0,(R4)
12579                                  ;STORE ACCUMULATOR 3
12580  054516  012704  054716          MOV     #HA3R,R4
12581  054522  172403                  LDD     AC3,AC0
```

```
12582  054524  174014                  STD     AC0,(R4)
12583                          ;STORE ACCUMULATOR 4
12584  054526  012704  054726          MOV     #HA4R,R4
12585  054532  172404                  LDD     AC4,AC0
12586  054534  174014                  STD     AC0,(R4)
12587                          ;STORE ACCUMULATOR 5
12588  054536  012704  054736          MOV     #HA5R,R4
12589  054542  172405                  LDD     AC5,AC0
12590  054544  174014                  STD     AC0,(R4)
12591  054546  000207                  RTS     PC
12592
12593                          ;COMPARE DATA LOADED WITH DATA READ.
12594  054550  012637  054622  HCMP:   MOV     (SP)+,@#HADR            ;SAVE RETURN ADDRESS.
12595  054554  012703  054626          MOV     #HA1W,R3
12596  054560  012704  054676          MOV     #HA1R,R4
12597  054564  012705  000024          MOV     #24,R5
12598  054570  022324          HCMP1:  CMP     (R3)+,(R4)+
12599  054572  001401                  BEQ     HCMP2
12600  054574  104000                  EMT                             ;
12601  054576  077504          HCMP2:  SOB     R5,HCMP1
12602  054600  000177  000016          JMP     @HADR
12603
12604                          ;CLEAR THE DATA OUTPUT BUFFER.
12605  054604  012704  054676  HCLR:   MOV     #HA1R,R4
12606  054610  012705  000024          MOV     #24,R5
12607  054614  005024          HCLR1:  CLR     (R4)+
12608  054616  077502                  SOB     R5,HCLR1
12609  054620  000207                  RTS     PC
12610
12611  054622  000000          HADR:   0
12612  054624  000000          HFLAG:  0
12613
12614  054626  000000  000000  000000  HA1W:   .WORD   0,0,0,0
12615  054634  000000
12616  054636  000000  000000  000000  HA2W:   .WORD   0,0,0,0
12617  054644  000000
12618  054646  000000  000000  000000  HA3W:   .WORD   0,0,0,0
12619  054654  000000
12620  054656  000000  000000  000000  HA4W:   .WORD   0,0,0,0
12621  054664  000000
12622  054666  000000  000000  000000  HA5W:   .WORD   0,0,0,0
12623  054674  000000
12624
12625  054676  000000  000000  000000  HA1R:   .WORD   0,0,0,0
12626  054704  000000
12627  054706  000000  000000  000000  HA2R:   .WORD   0,0,0,0
12628  054714  000000
12629  054716  000000  000000  000000  HA3R:   .WORD   0,0,0,0
12630  054724  000000
12631  054726  000000  000000  000000  HA4R:   .WORD   0,0,0,0
12632  054734  000000
12633  054736  000000  000000  000000  HA5R:   .WORD   0,0,0,0
12634  054744  000000
12635
12636  054746  073567  073567  073567  HDAT1:  .WORD   73567,73567,73567,73567
12637  054754  073567
```

```
12638  054756  063146  063146  063146  HDAT2:  .WORD   63146,63146,63146,63146
12639  054764  063146
12640  054766  010421  010421  010421  HDAT3:  .WORD   10421,10421,10421,10421
12641  054774  010421
12642  054776  031463  031463  031463  HDAT4:  .WORD   31463,31463,31463,31463
12643  055004  031463
12644  055006  042104  042104  042104  HDAT5:  .WORD   42104,42104,42104,42104
12645  055014  042104
12646
12647  055016                          HDONE:
12648  055016  004767  043156                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
12649                                                                  ;SEE IF THE USER HAS EXPRESSED
12650                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
12651                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12652                                                                  ;THE USER TYPED CONTROL G?).
12653
12654
12655                                  ;*****************************************************************************
12656                                  ;TEST 432       FSRC MODE 0 WITH ILLEGAL ACCUMULATOR TEST
12657                                  ;*****************************************************************************
12658  055022                          TS432:
12659  055022  170011                          SETD                    ;SET FD
12660  055024  012700  055304                  MOV     #SPAT10,R0      ;LOAD AC0
12661  055030  172410                          LDD     (R0),AC0
12662
12663  055032  012737  055210  000244          MOV     #SERR0,@#FPVECT ;USE OF THE NON-EXISTENT AC-
12664                                                                  ;CUMULATOR SHOULD RESULT IN
12665                                                                  ;A TRAP TO 244.
12666  055040  012700  000001                  MOV     #1,R0           ;A FAILURE IN THE FSRC FLOWS
12667                                                                  ;WILL RESULT IN AN ODD ADDRESS
12668  055044  012737  055116  000004          MOV     #SERR1,@#ERRVECT ;TRAP TO 4.
12669  055052  005003                          CLR     R3
12670
12671  055054  172407                  SX2:    LDD     AC7,AC0
12672  055056  170000                  SX3:    CFCC
12673  055060  005203                          INC     R3
12674  055062  005203                  SX4:    INC     R3
12675
12676  055064  012701  055314                  MOV     #SDAT00,R1      ;NO TRAP OCCURRED!!
12677  055070  174011                          STD     AC0,(R1)        ;SEE IF AC0 WAS MODIFIED.
12678
12679  055072  012701  055314                  MOV     #SDAT00,R1
12680  055076  012702  055304                  MOV     #SPAT10,R2
12681  055102  012703  000004                  MOV     #4,R3
12682  055106  022122                  SX5:    CMP     (R1)+,(R2)+
12683  055110  001401                          BEQ     SX6
12684  055112  104000                          EMT                     ;
12685  055114  077304                  SX6:    SOB     R3,SX5
12686  055116                          SERR1:
12687  055116  104000                          EMT                     ;
12688
12689                                  ;NOW TEST AC6.
12690  055120  170011                  SX7:    SETD
12691  055122  012700  055304                  MOV     #SPAT10,R0      ;LOAD AC0
12692  055126  172410                          LDD     (R0),AC0
12693  055130  012737  055260  000244          MOV     #SERR4,@#FPVECT
```

```
12694  055136  012700  000001              MOV     #1,R0
12695  055142  005003                      CLR     R3
12696
12697  055144  172406              SX8:    LDD     AC6,AC0
12698  055146  170000              SX9:    CFCC
12699  055150  005203                      INC     R3
12700  055152  005203              SX10:   INC     R3
12701
12702  055154  012701  055314            MOV     #SDAT00,R1
12703  055160  174011                      STD     AC0,(R1)                ;NO TRAP! GET AC0.
12704
12705  055162  012701  055314            MOV     #SDAT00,R1              ;WAS AC0 MODIFIED.
12706  055166  012702  055304            MOV     #SPAT10,R2
12707  055172  012703  000004            MOV     #4,R3
12708  055176  022122              SX11:   CMP     (R1)+,(R2)+
12709  055200  001401                      BEQ     SX12
12710  055202  104000                      EMT                             ;
12711  055204  077304              SX12:   SOB     R3,SX11
12712  055206  104000                      EMT                             ;
12713
12714                               ;TRAPPED TO 244.
12715  055210  021627  055056      SERR0:  CMP     (SP),#SX3               ;PC OF TRAP CORRECT?
12716  055214  001401                      BEQ     1$
12717  055216  104000                      EMT                             ;
12718  055220  012737  055120  055300 1$:  MOV     #SX7,@#SADR
12719  055226  022626              SERR10: CMP     (SP)+,(SP)+
12720  055230  005004                      CLR     R4
12721  055232  170204                      STFPS   R4                      ;IS FPS CORRECT?
12722  055234  022704  100200              CMP     #100200,R4
12723  055240  001326                      BNE     SERR1
12724
12725  055242  005004                      CLR     R4
12726  055244  170304                      STST    R4                      ;IS FEC CORRECT?
12727  055246  022704  000002              CMP     #2,R4
12728  055252  001321                      BNE     SERR1
12729  055254  000177  000020              JMP     @SADR
12730
12731  055260  021627  055146      SERR4:  CMP     (SP),#SX9
12732  055264  001401                      BEQ     1$
12733  055266  104000                      EMT                             ;
12734  055270  012737  055324  055300 1$:  MOV     #SDONE,@#SADR
12735  055276  000753                      BR      SERR10
12736
12737  055300  000000              SADR:   0
12738  055302  177777                      -1
12739  055304  010421              SPAT10: 10421
12740  055306  021042              SPAT11: 21042
12741  055310  031463              SPAT12: 31463
12742  055312  042104              SPAT13: 42104
12743
12744  055314  000000              SDAT00: 0
12745  055316  000000              SDAT01: 0
12746  055320  000000              SDAT02: 0
12747  055322  000000              SDAT03: 0
12748
12749  055324                      SDONE:
```

```
12750  055324  004767  042650                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
12751                                                                 ;SEE IF THE USER HAS EXPRESSED
12752                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
12753                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12754                                                                 ;THE USER TYPED CONTROL G?).
12755
12756          ;*************************************************************************************
12757          ;TEST 433       FSRC MODE 2 TEST
12758          ;*************************************************************************************
12759  055330  TS433:
12760  055330  J1:
12761  055330  170011                        SETD                    ;SET DOUBLE MODE
12762
12763  055332  012700  055460                MOV     #JDAT0,R0
12764  055336  172410                        LDD     (R0),AC0        ;LOAD AC0=ALL 1
12765
12766  055340  012700  055440                MOV     #JDATI0,R0
12767  055344  005003                        CLR     R3
12768
12769  055346  172420          J2:           LDD     (R0)+,AC0       ;TEST INSTRUCTION
12770  055350  005203          J3:           INC     R3
12771  055352  005203          J4:           INC     R3
12772
12773  055354  012701  055450                MOV     #JDATO0,R1
12774  055360  174011                        STD     AC0,(R1)        ;PICK UP RESULTS
12775
12776  055362  020027  055430                CMP     R0,#JBUF0       ;WAS AN AUTO
12777  055366  001001                        BNE     1$
12778  055370  104000                        EMT                     ;
12779  055372  012702  055440  1$:           MOV     #JDATI0,R2      ;IS DATA CORRECT?
12780  055376  012703  055450                MOV     #JDATO0,R3
12781  055402  012704  000004                MOV     #4,R4
12782  055406  022223          J5:           CMP     (R2)+,(R3)+
12783  055410  001401                        BEQ     J6
12784  055412  104000                        EMT                     ;
12785  055414  077404          J6:           SOB     R4,J5
12786
12787  055416  022700  055450                CMP     #JDATI0+10,R0   ;WAS R0 INCREM.
12788  055422  001401                        BEQ     J7
12789  055424  104000                        EMT                     ;
12790  055426  000420          J7:           BR      JDONE
12791
12792  055430  010421          JBUF0:        .WORD   010421
12793  055432  021042          JBUF1:                021042
12794  055434  042104          JBUF2:                042104
12795  055436  031463          JBUF3:                031463
12796
12797  055440  052525          JDATI0:               052525
12798  055442  114631          JDATI1:               114631
12799  055444  063146          JDATI2:               063146
12800  055446  073567          JDATI3:               073567
12801
12802  055450  000000          JDATO0:               0
12803  055452  000000          JDATO1:               0
12804  055454  000000          JDATO2:               0
12805  055456  000000          JDATO3:               0
```

```
12806
12807  055460  177777                  JDAT0:          -1
12808  055462  177777                  JDAT1:          -1
12809  055464  177777                  JDAT2:          -1
12810  055466  177777                  JDAT3:          -1
12811
12812
12813  055470                          JDONE:
12814  055470  004767  042504                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
12815                                                                  ;SEE IF THE USER HAS EXPRESSED
12816                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
12817                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12818                                                                  ;THE USER TYPED CONTROL G?).
12819
12820
12821                                   ;*****************************************************************************
12822                                   ;TEST 434       FSRC MODE 4 TEST
12823                                   ;*****************************************************************************
12824  055474                          TS434:
12825  055474  170011                          SETD                    ;SET DOUBLE MODE
12826
12827  055476  012700  055624                  MOV     #KPAT0,R0
12828  055502  172410                          LDD     (R0),AC0        ;LOAD A DEFAULT
12829                                                                  ;PATTERN INTO AC0
12830  055504  012700  055604                  MOV     #KBUF0,R0
12831  055510  005003                          CLR     R3
12832  055512  172440                  KX2:    LDD     -(R0),AC0       ;TEST INSTRUCTION
12833  055514  005203                  KX3:    INC     R3
12834  055516  005203                  KX4:    INC     R3
12835
12836  055520  012701  055614                  MOV     #KDAT00,R1
12837  055524  174011                          STD     AC0,(R1)        ;PICK UP THE RESULT
12838
12839  055526  020027  055614                  CMP     R0,#KBUF0+10    ;WAS AN AUTO
12840  055532  001001                          BNE     1$
12841  055534  104000                          EMT                     ;
12842  055536  012702  055574          1$:     MOV     #KDATI0,R2      ;IS DATA CORRECT?
12843  055542  012703  055614                  MOV     #KDAT00,R3
12844  055546  012704  000004                  MOV     #4,R4
12845  055552  022223                  KX5:    CMP     (R2)+,(R3)+
12846  055554  001401                          BEQ     KX6
12847  055556  104000                          EMT                     ;
12848  055560  077404                  KX6:    SOB     R4,KX5
12849
12850  055562  022700  055574                  CMP     #KBUF0-10,R0    ;WAS R0 DECREMENTED
12851  055566  001401                          BEQ     KX7
12852  055570  104000                          EMT                     ;
12853  055572  000420                  KX7:    BR      KDONE
12854
12855  055574  052525                  KDATI0: .WORD   052525
12856  055576  114631                  KDATI1:         114631
12857  055600  063140                  KDATI2:         063140
12858  055602  073567                  KDATI3:         073567
12859
12860  055604  010421                  KBUF0:          010421
12861  055606  031463                  KBUF1:          031463
```

```
12862  055610  042104                  KBUF2:          042104
12863  055612  021042                  KBUF3:          021042
12864
12865  055614  000000                  KDAT00:         0
12866  055616  000000                  KDAT01:         0
12867  055620  000000                  KDAT02:         0
12868  055622  000000                  KDAT03:         0
12869
12870  055624  177777                  KPAT0:          -1
12871  055626  177777                  KPAT1:          -1
12872  055630  177777                  KPAT2:          -1
12873  055632  177777                  DPAT3:          -1
12874
12875  055634                          KDONE:
12876  055634  004767  042340                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
12877                                                                  ;SEE IF THE USER HAS EXPRESSED
12878                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
12879                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
12880                                                                  ;THE USER TYPED CONTROL G?).
12881
12882
12883                                   ;*****************************************************************************
12884                                   ;TEST 435       FSRC MODE 2, WITH FD=0, TEST
12885                                   ;*****************************************************************************
12886  055640                          TS435:
12887  055640                          L1:
12888  055640  170011                          SETD                    ;SET DOUBLE MODE
12889
12890  055642  012700  055766                  MOV     #LPAT10,R0
12891  055646  172410                          LDD     (R0),AC0        ;LOAD AC0
12892
12893  055650  012700  056010                  MOV     #LDAT10,R0      ;SET UP THE INPUT
12894  055654  012701  055776                  MOV     #LPAT20,R1      ;DATA
12895  055660  012702  000004                  MOV     #4,R2
12896
12897  055664  012120                  1$:     MOV     (R1)+,(R0)+
12898  055666  077202                          SOB     R2,1$
12899
12900  055670  012700  056010                  MOV     #LDAT10,R0
12901  055674  005003                          CLR     R3
12902  055676  170001                          SETF                    ;CLEAR FD.
12903
12904  055700  172420                  L2:     LDF     (R0)+,AC0
12905  055702  005203                  L3:     INC     R3
12906
12907  055704                          L4:
12908  055704  170011                          SETD                    ;SET FD
12909
12910  055706  012701  056022                  MOV     #LDAT00,R1
12911  055712  174011                          STD     AC0,(R1)        ;PICK UP RESULTS
12912
12913  055714  020027  056014                  CMP     R0,#LDAT12      ;WAS R0 INCREMENTED
12914  055720  001401                          BEQ     1$
12915  055722  104000                          EMT                     ;
12916  055724  012737  177777  056014  1$:     MOV     #-1,@#LDAT12
12917  055732  012737  177777  056016          MOV     #-1,@#LDAT13
```

```
 12918  055740  012702  056010                MOV     #LDATI0,R2      ;IS DATA CORRECT
 12919  055744  012703  056022                MOV     #LDATO0,R3
 12920  055750  012704  000004                MOV     #4,R4
 12921
 12922  055754  022223              L5:       CMP     (R2)+,(R3)+
 12923  055756  001401                        BEQ     L6
 12924  055760  104000                        EMT                     ;
 12925  055762  077404              L6:       SOB     R4,L5
 12926  055764  000422                        BR      LDONE
 12927
 12928  055766  177777              LPAT10: .WORD   -1
 12929  055770  177777              LPAT11:         -1
 12930  055772  177777              LPAT12:         -1
 12931  055774  177777              LPAT13:         -1
 12932
 12933  055776  052525              LPAT20:         052525
 12934  056000  114631              LPAT21:         114631
 12935  056002  063142              LPAT22:         063142
 12936  056004  073567              LPAT23:         073567
 12937  056006  000001                      .WORD   000001
 12938  056010  000000              LDATI0:         0
 12939  056012  000000              LDATI1:         0
 12940  056014  000000              LDATI2:         0
 12941  056016  000000              LDATI3:         0
 12942  056020  000001                      .WORD   00001
 12943  056022  000000              LDATO0:         0
 12944  056024  000000              LDATO1:         0
 12945  056026  000000              LDATO2:         0
 12946  056030  000000              LDATO3:         0
 12947
 12948  056032                      LDONE:
 12949  056032  004767  042142              JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
 12950                                                              ;SEE IF THE USER HAS EXPRESSED
 12951                                                              ;THE DESIRE TO CHANGE THE SOFTWARE
 12952                                                              ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
 12953                                                              ;THE USER TYPED CONTROL G?).
 12954
 12955
 12956                              ;*********************************************************************************************
 12957                              ;TEST 436       FSRC MODE 2 WITH GR7, IMMEDIATE MODE, TEST
 12958                              ;*********************************************************************************************
 12959  056036                      TS436:
 12960
 12961  056036                      M1:
 12962  056036  170011                      SETD
 12963
 12964  056040  012700  056140              MOV     #MPAT10,R0
 12965  056044  172410                      LDD     (R0),AC0        ;LOAD BACKROUND
                                                                     ;PATTERN INTO AC0.
 12966
 12967  056046  005004                      CLR     R4
 12968  056050  012737  056076  000004      MOV     #MERR3,@#ERRVECT
 12969
 12970  056056  172427  000000      M15:    LDD     #0,AC0          ;TEST INSTRUCTION
 12971          056060                      .=.-2           ;EFFECTIVELY: 05204 IS PUT IN THE FIRST
 12972  056060  005204                      .WORD   5204    ;16 BIT WORD, OR THE "EXP-FRACTION" WORD.
 12973  056062  005204              M2:     INC     R4              ;NOTE THAT
```

```
12974  056064  005204
12975  056066  005204
12976  056070  020427  000003
12977  056074  001401
12978  056076
12979  056076  104000
12980  056100  012700  056160
12981  056104  174010
12982
12983  056106  012700  056160
12984  056112  022720  005204
12985  056116  001401
12986  056120  104000
12987  056122  012701  000003
12988  056126  005720
12989  056130  001401
12990  056132  104000
12991  056134  077104
12992  056136  000414
12993
12994  056140  177777
12995  056142  177777
12996  056144  177777
12997  056146  177777
12998
12999  056150  005204
13000  056152  005204
13001  056154  005204
13002  056156  005204
13003
13004  056160  000000
13005  056162  000000
13006  056164  000000
13007  056166  000000
13008
13009  056170
13010  056170  004767  042004
13011
13012
13013
13014
13015
13016
13017
13018
13019
13020  056174
13021
13022  056174
13023  056174  170011
13024
13025  056176  012700  056324
13026  056202  172410
13027
13028  056204  012700  056312
13029  056210  005003
```

```
M3:     INC     R4              ;005204=INC R4
M4:     INC     R4
        CMP     R4,#3           ;SEE IF THE PC
        BEQ     M8
MERR3:
        EMT                     ;
M8:     MOV     #MDAT00,R0
        STD     AC0,(R0)        ;GET THE DATA

        MOV     #MDAT00,R0
        CMP     #5204,(R0)+     ;IS THE DATA CORRECT?
        BEQ     M5
        EMT                     ;
M5:     MOV     #3,R1
M6:     TST     (R0)+
        BEQ     M7
        EMT                     ;
M7:     SOB     R1,M6
        BR      MDONE

MPAT10:         -1
MPAT11:         -1
MPAT12:         -1
MPAT13:         -1

MPAT20:         5204
MPAT21:         5204
MPAT22:         5204
MPAT23:         5204

MDAT00:         0
MDAT01:         0
MDAT02:         0
MDAT03:         0

MDONE:
        JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                ;SEE IF THE USER HAS EXPRESSED
                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                ;THE USER TYPED CONTROL G?).

;*******************************************************************************
;TEST 437       FSRC MODE 3 TEST
;*******************************************************************************
TS437:

N1:
        SETD                    ;SET FD MODE

        MOV     #NPAT10,R0
        LDD     (R0),AC0        ;LOAD AC0 WITH A DEFAULT
                                ;PATTERN
        MOV     #NPAT20,R0
        CLR     R3
```

```
13030  056212  012737  056264  000004          MOV     #NERR0,@#ERRVECT        ;IF A FAILURE OCCURS
13031                                                                           ;IN THE FSRC FLOWS AN
13032                                                                           ;ODD TRAP TO 4 COULD OCCUR
13033  056220  172430                  N2:     LDD     @(R0)+,AC0              ;TEST INSTRUCTION.
13034  056222  005203                  N3:     INC     R3
13035  056224  005203                  N4:     INC     R3
13036
13037  056226  012701  056272                  MOV     #NDAT00,R1
13038  056232  174011                          STD     AC0,(R1)                ;GET THE DATA
13039
13040  056234  020027  056314                  CMP     R0,#NPAT20+2            ;WAS R0 INCREMENTED
13041  056240  001401                          BEQ     N12
13042  056242  104000                          EMT                             ;
13043  056244  012702  056272          N12:    MOV     #NDAT00,R2              ;DATA CORRECT
13044  056250  012703  056334                  MOV     #NDAT10,R3
13045  056254  012704  000004                  MOV     #4,R4
13046  056260  022223                  N13:    CMP     (R2)+,(R3)+
13047  056262  001401                          BEQ     N14
13048  056264                          NERR0:
13049  056264  104000                          EMT                             ;
13050  056266  077404                  N14:    SOB     R4,N13
13051  056270  000425                          BR      NDONE
13052
13053  056272  000000                  NDAT00: .WORD   0
13054  056274  000000                  NDAT01:         0
13055  056276  000000                  NDAT02:         0
13056  056300  000000                  NDAT03:         0
13057
13058  056302  052525  052525  052525          .WORD   52525,52525,52525,52525
13059  056310  052525
13060  056312  056334                  NPAT20: .WORD   NDAT10
13061  056314  070707                  NPAT21:         070707
13062  056316  070707                  NPAT22:         070707
13063  056320  070707                  NPAT23:         070707
13064  056322  000001                          .WORD   1
13065  056324  177777                  NPAT10: .WORD   -1
13066  056326  177777                  NPAT11:         -1
13067  056330  177777                  NPAT12:         -1
13068  056332  177777                  NPAT13:         -1
13069
13070  056334  010421                  NDAT10: .WORD   010421
13071  056336  021042                  NDAT11:         021042
13072  056340  031463                  NDAT12:         031463
13073  056342  042104                  NDAT13:         042104
13074
13075  056344                          NDONE:
13076  056344  004767  041630                  JSR     PC,..RSET               ;GO INITIALIZE THE FPS AND STACK; AND
13077                                                                           ;SEE IF THE USER HAS EXPRESSED
13078                                                                           ;THE DESIRE TO CHANGE THE SOFTWARE
13079                                                                           ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
13080                                                                           ;THE USER TYPED CONTROL G?).
13081
13082
13083                          ;*****************************************************************************
13084                          ;TEST 440       FSRC MODE 5 TEST
13085                          ;*****************************************************************************
```

```
13086  056350                          TS440:
13087
13088  056350                          01:
13089  056350  170011                          SETD                    ;SET FD MODE
13090
13091  056352  012700  056500                  MOV     #OPAT10,R0
13092  056356  172410                          LDD     (R0),AC0        ;LOAD AC0 WITH A
13093                                                                   ;DEFAULT PATTERN.
13094  056360  012700  056466                  MOV     #OPAT21,R0
13095  056364  005003                          CLR     R3
13096  056366  012737  056416  000004         MOV     #OERR0,@#ERRVEC ;IF A FAILURE
13097                                                                   ;OCCURS IN THE FSRC
13098                                                                   ;FLOWS AN ODD ADDR.
13099                                                                   ;TRAP TO 4 MAY OCCUR.
13100  056374  172450                  02:     LDD     @-(R0),AC0      ;TEST INSTRUCTION
13101  056376  005203                  03:     INC     R3
13102  056400  005203                  04:     INC     R3
13103
13104  056402  012701  056446                  MOV     #ODAT00,R1
13105  056406  174011                          STD     AC0,(R1)        ;GET THE DATA
13106
13107  056410  020027  056464                  CMP     R0,#OPAT20      ;WAS R0 DECREMENTED
13108  056414  001401                          BEQ     012
13109  056416                          OERR0:
13110  056416  104000                          EMT                     ;
13111  056420  012702  056446          012:    MOV     #ODAT00,R2      ;DATA CORRECT?
13112  056424  012703  056510                  MOV     #ODATI0,R3
13113  056430  012704  000004                  MOV     #4,R4
13114  056434  022223                  013:    CMP     (R2)+,(R3)+
13115  056436  001401                          BEQ     014
13116  056440  104000                          EMT                     ;
13117  056442  077404                  014:    SOB     R4,013
13118  056444  000425                          BR      ODONE
13119
13120  056446  000000                  ODAT00: .WORD   0
13121  056450  000000                  ODAT01:         0
13122  056452  000000                  ODAT02:         0
13123  056454  000000                  ODAT03:         0
13124
13125  056456  052525  052525  052525          .WORD   52525,52525,52525
13126  056464  056510                  OPAT20: .WORD   ODATI0
13127  056466  070707                  OPAT21: 070707
13128  056470  070707                  OPAT22: 070707
13129  056472  070707                  OPAT23: 070707
13130  056474  070707                  OPAT24: 070707
13131  056476  000001                          .WORD   1
13132  056500  177777                  OPAT10: .WORD   -1
13133  056502  177777                  OPAT11:         -1
13134  056504  177777                  OPAT12:         -1
13135  056506  177777                  OPAT13:         -1
13136
13137  056510  073567                  ODATI0: .WORD   73567
13138  056512  004210                  ODATI1:         004210
13139  056514  114631                  ODATI2:         114631
13140  056516  125252                  ODATI3:         125252
13141
```

```
13142  056520
13143  056520  004767  041454          ODONE:  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
13144                                                                  ;SEE IF THE USER HAS EXPRESSED
13145                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
13146                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
13147                                                                  ;THE USER TYPED CONTROL G?).
13148
13149
13150          ;*****************************************************************************************
13151          ;TEST 441       FSRC MODE 6 TEST
13152          ;*****************************************************************************************
13153  056524  TS441:
13154
13155  056524  P1:
13156  056524  170011                  SETD                            ;SET FD MODE
13157
13158  056526  012700  056612          MOV     #PPAT10,R0
13159  056532  172410                  LDD     (R0),AC0                ;LOAD A DEFAULT PATTERN
13160                                                                  ;INTO AC0
13161  056534  012700  056361          MOV     #PDATI0-241,R0          ;COULD OCCUR.
13162
13163  056540  172460  000241  P2:     LDD     241(R0),AC0
13164          056542          P3=P2+2
13165
13166  056544  012701  056632  P4:     MOV     #PDAT00,R1
13167  056550  174011                  STD     AC0,(R1)                ;GET THE DATA
13168  056552  012703  000004          MOV     #4,R3
13169  056556  012702  056622          MOV     #PDATI0,R2
13170  056562  012701  056632          MOV     #PDAT00,R1
13171  056566  022221  P5:     CMP     (R2)+,(R1)+             ;CHECK THE DATA
13172  056570  001401                  BEQ     2$
13173  056572  104000                  EMT                             ;
13174  056574  077304          2$:     SOB     R3,P5
13175  056576  022700  056361          CMP     #PDATI0-241,R0          ;R0 CORRECT?
13176  056602  001401                  BEQ     1$
13177  056604  104000                  EMT                             ;
13178  056606  000137  056642  1$:     JMP     @#PDONE
13179  056612  177777  PPAT10: .WORD   -1
13180  056614  177777  PPAT11:         -1
13181  056616  177777  PPAT12:         -1
13182  056620  177777  PPAT13:         -1
13183
13184  056622  010421  PDATI0: .WORD   010421
13185  056624  031463  PDATI1:         031463
13186  056626  052525  PDATI2:         052525
13187  056630  073567  PDATI3:         073567
13188
13189  056632  000000  PDAT00: .WORD   0
13190  056634  000000  PDAT01:         0
13191  056636  000000  PDAT02:         0
13192  056640  000000  PDAT03:         0
13193
13194  056642  PDONE:
13195  056642  004767  041332          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
13196                                                          ;SEE IF THE USER HAS EXPRESSED
13197                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
```

```
13198                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
13199                                                   ;THE USER TYPED CONTROL G?).
13200
13201
13202                           ;*****************************************************************************
13203                           ;TEST 442       FSRC MODE 7 TEST
13204                           ;*****************************************************************************
13205  056646                   TS442:
13206
13207  056646                   Q1:
13208  056646  170011                   SETD
13209
13210  056650  012700  056734           MOV     #QPAT10,R0
13211  056654  172410                   LDD     (R0),AC0        ;LOAD A DEFAULT
13212                                                           ;PATTERN INTO AC0
13213  056656  012700  056503           MOV     #QPAT20-241,R0
13214
13215  056662  172470  000241   Q2:     LDD     @241(R0),AC0
13216          056664           Q3=Q2+2
13217
13218  056666  012701  056754   Q4:     MOV     #QDAT00,R1
13219  056672  174011                   STD     AC0,(R1)        ;GET THE DATA
13220
13221  056674  012703  000004           MOV     #4,R3
13222  056700  012704  056754           MOV     #QDAT00,R4
13223  056704  012705  056764           MOV     #QDAT10,R5
13224  056710  022425           Q5:     CMP     (R4)+,(R5)+     ;CHECK THE DATA
13225  056712  001401                   BEQ     2$
13226  056714  104000                   EMT                     ;
13227  056716  077304           2$:     SOB     R3,Q5
13228
13229  056720  022700  056503           CMP     #QPAT20-241,R0  ;CHECK R0.
13230  056724  001401                   BEQ     1$
13231  056726  104000                   EMT                     ;
13232  056730  000137  056774   1$:     JMP     @#QDONE
13233
13234  056734  177777           QPAT10: .WORD   -1
13235  056736  177777           QPAT11:         -1
13236  056740  177777           QPAT12:         -1
13237  056742  177777           QPAT13:         -1
13238
13239  056744  056764           QPAT20: .WORD   QDAT10
13240  056746  052525           QPAT21:         52525
13241  056750  052525           QPAT22:         52525
13242  056752  052525           QPAT23:         52525
13243
13244  056754  000000           QDAT00: .WORD   0
13245  056756  000000           QDAT01:         0
13246  056760  000000           QDAT02:         0
13247  056762  000000           QDAT03:         0
13248
13249  056764  073567           QDAT10: .WORD   073567
13250  056766  052525           QDAT11: .WORD   052525
13251  056770  031463           QDAT12: .WORD   031463
13252  056772  010421           QDAT13: .WORD   010421
13253
```

```
13254  056774
13255  056774  004767  041200                  QDONE:
                                                       JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
                                                                               ;SEE IF THE USER HAS EXPRESSED
                                                                               ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                               ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                               ;THE USER TYPED CONTROL G?).
13256
13257                                           ;*************************************************************************
13258                                           ;TEST 443       (BUT EZBT Y8),(BUT ENBT) AND (BUT FIUV) TEST
13259                                           ;*************************************************************************
13260                                           TS443:
13261                                                   CLR     @#UFLAG
13262                                                   MOV     #UPAT00,R0      ;SET UP AC#0 DATA.
13263                                                   MOV     #4,R1
13264  057000                                   U0:    MOV     #-1,(R0)+
13265  057000  005037  057552                          SOB     R1,U0
13266  057004  012700  057502
13267  057010  012701  000004
13268  057014  012720  177777
13269  057020  077103
13270
13271  057022  012737  000033  057554                  MOV     #033,@#UTMP1
13272  057030  012737  000023  057556                  MOV     #023,@#UTMP2
13273  057036  012737  057122  000244           U1:    MOV     #UERR0,@#FPVECT ;IN CASE (BUT FIUV FAILS)
13274  057044  012700  000200                          MOV     #200,R0
13275  057050  170100                                  LDFPS   R0
13276  057052  012700  057502                          MOV     #UPAT00,R0      ;LOAD AC0
13277  057056  172410                                  LDD     (R0),AC0
13278  057060  013737  057554  057560                  MOV     @#UTMP1,@#UROM1
13279  057066  012737  000001  057562                  MOV     #001,@#UROM2
13280  057074  012737  000254  057564                  MOV     #254,@#UROM3
13281
13282  057102  012700  057512                          MOV     #UPAT10,R0      ;LOAD 0 INTO AC0
13283  057106  172410                           U2:    LDD     (R0),AC0
13284  057110  012704  000204                          MOV     #204,R4         ;SEE IF FPS IS CORRECT
13285  057114  170205                                  STFPS   R5
13286
13287  057116  020405                                  CMP     R4,R5
13288  057120  001401                                  BEQ     U3
13289  057122                                   UERR0:
13290  057122  104000                                  EMT                     ;
13291  057124  012700  000200                   U3:    MOV     #200,R0
13292  057130  170100                                  LDFPS   R0
13293
13294  057132  012700  057502                          MOV     #UPAT00,R0      ;LOAD AC0
13295  057136  172410                                  LDD     (R0),AC0
13296  057140  013737  057556  057560                  MOV     @#UTMP2,@#UROM1
13297  057146  012737  000003  057562                  MOV     #003,@#UROM2
13298  057154  012737  000054  057564                  MOV     #054,@#UROM3
13299
13300  057162  012700  057522                          MOV     #UPAT20,R0      ;LOAD A POSITIVE NUMBER
13301                                                                           ;INTO AC0
13302  057166  172410                           U4:    LDD     (R0),AC0
13303  057170  012704  000200                          MOV     #200,R4         ;FPS CORRECT?
13304  057174  170205                                  STFPS   R5
13305
13306  057176  020405                                  CMP     R4,R5
13307  057200  001401                                  BEQ     U5
13308  057202  104000                                  EMT                     ;
13309  057204  012700  000200                   U5:    MOV     #200,R0
```

```
13310  057210  170100                          LDFPS   R0
13311  057212  012700  057502                  MOV     #UPAT00,R0      ;LOAD AC0
13312  057216  172410                          LDD     (R0),AC0
13313  057220  013737  057556  057560          MOV     @#UTMP2,@#UROM1
13314  057226  012737  000403  057562          MOV     #403,@#UROM2
13315  057234  012737  000056  057564          MOV     #056,@#UROM3
13316  057242  012700  057532                  MOV     #UPAT30,R0      ;LOAD A NEGATIUE
13317                                                                  ;NUMBER INTO AL0
13318  057246  172410                  U6:     LDD     (R0),AC0
13319  057250  012704  000210                  MOV     #210,R4         ;FPS CORRECT
13320  057254  170205                          STFPS   R5
13321  057256  020405                          CMP     R4,R5
13322  057260  001401                          BEQ     U7
13323  057262  104000                          EMT                     ;
13324  057264  012700  000200          U7:     MOV     #200,R0
13325  057270  170100                          LDFPS   R0
13326  057272  012700  057502                  MOV     #UPAT00,R0      ;LOAD AC0
13327  057276  172410                          LDD     (R0),AC0
13328  057300  013737  057554  057560          MOV     @#UTMP1,@#UROM1
13329  057306  012737  000401  057562          MOV     #401,@#UROM2
13330  057314  012737  000256  057564          MOV     #256,@#UROM3
13331  057322  012700  057542                  MOV     #UPAT40,R0      ;LOAD -0 INTO AC0
13332  057326  172410                  U10:    LDD     (R0),AC0
13333  057330  000240                  U11:    NOP                     ;TRAP FROM HERE IF
13334  057332  012704  000214                  MOV     #214,R4         ;SEE IF FPS IS CORRECT.
13335  057336  170205                          STFPS   R5
13336  057340  020405                          CMP     R4,R5
13337  057342  001401                          BEQ     U12
13338  057344  104000                          EMT                     ;
13339  057346  005737  057552          U12:    TST     @#UFLAG ;SEE IF ALL THE PATTERNS
13340  057352  001021                          BNE     U14             ;HAVE BEEN TEST WITH
13341                                                                  ;BOTH AC NOT EQUAL TO 0 AND AC=0
13342  057354  012700  057502                  MOV     #UPAT00,R0      ;IF NOT GO BACK AND
13343  057360  012701  000004                  MOV     #4,R1           ;CHECK THEM WITH AC=0
13344  057364  005020                  U13:    CLR     (R0)+
13345  057366  077102                          SOB     R1,U13
13346  057370  012737  177777  057552          MOV     #-1,@#UFLAG
13347  057376  012737  000233  057554          MOV     #233,@#UTMP1
13348  057404  012737  000223  057556          MOV     #223,@#UTMP2
13349  057412  000137  057036                  JMP     @#U1
13350                          ;NOW SEE IF A TRAP CAN BE FORCED BY SETTING FIUV AND LOADING -0
13351  057416  012737  057454  000244  U14:    MOV     #UERR3,@#FPVECT
13352  057424  012700  004200                  MOV     #4200,R0        ;SET FD AND FIUV
13353  057430  170100                          LDFPS   R0
13354  057432  012700  057502                  MOV     #UPAT00,R0      ;SET UP AC0
13355  057436  172410                          LDD     (R0),AC0
13356  057440  012700  057542                  MOV     #UPAT40,R0      ;LOAD -0
13357  057444  172410                  U15:    LDD     (R0),AC0        ;SHOULD TRAP TO 244
13358  057446  170000                  U16:    CFCC
13359  057450  000240                          NOP
13360  057452  104000                          EMT                     ;
13361
13362                          ;INTERRUPT HERE WHEN FIUV SET AND ATTEMPTED TO LOAD-0
13363  057454  021627  057446          UERR3:  CMP     (SP),#U16
13364  057460  001401                          BEQ     1$
13365  057462  104000                          EMT                     ;
```

```
13366  057464  022626                 1$:     CMP     (SP)+,(SP)+
13367  057466  005000                         CLR     R0
13368  057470  170300                         STST    R0              ;GET FEC.
13369  057472  022700  000014               CMP     #14,R0          ;CORRECT
13370  057476  001433                       BEQ     UDONE
13371  057500  104000                       EMT                     ;
13372  057502  000000                 UPAT00: .WORD   0
13373  057504  000000                 UPAT01:         0
13374  057506  000000                 UPAT02:         0
13375  057510  000000                 UPAT03:         0
13376
13377  057512  000000                 UPAT10: .WORD   0               ;0
13378  057514  000000                 UPAT11:         0
13379  057516  000000                 UPAT12:         0
13380  057520  000000                 UPAT13:         0
13381
13382  057522  010421                 UPAT20: .WORD   010421          ;POS NUM
13383  057524  114631                 UPAT21:         114631
13384  057526  125252                 UPAT22:         125252
13385  057530  177777                 UPAT23:         177777
13386
13387  057532  114631                 UPAT30:         114631          ;NEG NUM
13388  057534  135673                 UPAT31:         135673
13389  057536  146314                 UPAT32:         146314
13390  057540  167356                 UPAT33:         167356
13391
13392  057542  100000                 UPAT40:         100000          ;NEG ZERO
13393  057544  000000                 UPAT41:         0
13394  057546  000000                 UPAT42:         0
13395  057550  000000                 UPAT43:         0
13396
13397  057552  000000                 UFLAG:  .WORD   0
13398  057554  000000                 UTMP1:          0
13399  057556  000000                 UTMP2:          0
13400  057560  000000                 UROM1:          0
13401  057562  000000                 UROM2:          0
13402  057564  000000                 UROM3:          0
13403  057566                         UDONE:
13404
13405
13406                                  ;*****************************************************************************************
13407                                  ;TEST 444       ADDF,ADDD,SUBF AND SUBD WITH FSRC=AC=0 TEST
13408                                  ;*****************************************************************************************
13409  057566                         TS444:
13410  057566  012700  000200                 MOV     #200,R0
13411  057572  170100                         LDFPS   R0              ;SET DOUBLE MODE
13412  057574  012700  060116                 MOV     #WPAT00,R0      ;LOAD AC0=:
13413  057600  172410                         LDD     (R0),AC0
13414  057602  012700  060116                 MOV     #WPAT00,R0
13415  057606  172010                 W2:     ADDD    (R0),AC0        ;TEST INSTRUCTION. ADD ITSELF
13416  057610  170205                         STFPS   R5              ;GET FPS
```

```
13417  057612  170011
13418  057614  012700  060116          SETD                    ;SET DOUBLE MODE
13419  057620  174010                  MOV     #WPAT00,R0
13420  057622  012701  060116          STD     AC0,(R0)        ;GET THE RESULT
13421  057626  012702  000004          MOV     #WPAT00,R1
13422  057632  022021                  MOV     #4,R2
13423  057634  001401          W3:     CMP     (R0)+,(R1)+     ;IS RESULT CORRECT
13424  057636  104000                  BEQ     W4
13425  057640  077204                  EMT                     ;
13426  057642  022705  000204  W4:     SOB     R2,W3
13427  057646  001401                  CMP     #204,R5         ;IS FPS CORRECT
13428  057650  104000                  BEQ     W5
13429  057652  012700  000200          EMT                     ;
13430  057656  170100          W5:     MOV     #200,R0
13431  057660  012700  060116          LDFPS   R0              ;SET DOUBLE MODE
13432  057664  172410                  MOV     #WPAT00,R0      ;LOAD AC0=0
13433  057666  005000                  LDD     (R0),AC0
13434  057670  170100                  CLR     R0
13435  057672  012700  060116          LDFPS   R0              ;GO TO FLOATING MODE
13436  057676  172010                  MOV     #WPAT00,R0
13437  057700  170205          W6:     ADDF    (R0),AC0        ;TEST INSTRUCTION
13438  057702  170011                  STFPS   R5              ;GET FPS
13439  057704  012700  060116          SETD                    ;RESET TO DOUBLE MODE
13440  057710  174010                  MOV     #WPAT00,R0
13441  057712  012701  060116          STD     AC0,(R0)        ;GET THE RESULT
13442  057716  012702  000004          MOV     #WPAT00,R1
13443  057722  022021                  MOV     #4,R2
13444  057724  001401          W7:     CMP     (R0)+,(R1)+     ;WAS THE RESULT
13445  057726  104000                  BEQ     W10
13446  057730  077204                  EMT                     ;
13447  057732  022705  000004  W10:    SOB     R2,W7
13448  057736  001401                  CMP     #4,R5           ;WAS FPS CORRECT
13449  057740  104000                  BEQ     W11
13450  057742  012700  000200          EMT                     ;
13451  057746  170100          W11:    MOV     #200,R0
13452  057750  012700  060116          LDFPS   R0              ;SET DOUBLE MODE
13453  057754  172410                  MOV     #WPAT00,R0      ;LOAD AC0=0
13454  057756  012700  060116          LDD     (R0),AC0
13455  057762  173010                  MOV     #WPAT00,R0
13456  057764  170205          W12:    SUBD    (R0),AC0        ;TEST INSTRUCTION
13457  057766  170011                  STFPS   R5              ;GET FPS
13458  057770  012700  060116          SETD                    ;SET DOUBLE MODE
13459  057774  174010                  MOV     #WPAT00,R0
13460  057776  012701  060116          STD     AC0,(R0)        ;GET THE RESULT
13461  060002  012702  000004          MOV     #WPAT00,R1
13462  060006  022021                  MOV     #4,R2
13463  060010  001401          W13:    CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
13464  060012  104000                  BEQ     W14
13465  060014  077204                  EMT                     ;
13466  060016  022705  000204  W14:    SOB     R2,W13
13467  060022  001401                  CMP     #204,R5         ;IS FPS CORRECT?
13468  060024  104000                  BEQ     W15
13469  060026  012700  000200          EMT                     ;
13470  060032  170100          W15:    MOV     #200,R0
13471  060034  012700  060116          LDFPS   R0              ;SET DOUBLE MODE
13472  060040  172410                  MOV     #WPAT00,R0      ;LOAD AC0=0
                                       LDD     (R0),AC0
```

```
13473  060042  005000
13474  060044  170100
13475  060046  012700  060116
13476  060052  173010
13477  060054  170205
13478  060056  170011
13479  060060  012700  060116
13480  060064  174010
13481  060066  012701  060116
13482  060072  012702  000004
13483  060076  022021
13484  060100  001401
13485  060102  104000
13486  060104  077204
13487  060106  022705  000004
13488  060112  001411
13489  060114  104000
13490
13491  060116  000000
13492  060120  000000
13493  060122  000000
13494  060124  000000
13495
13496  060126  000000
13497  060130  000000
13498  060132  000000
13499  060134  000000
13500
13501  060136
13502  060136  004767  040036
13503
13504
13505
13506
13507
13508
13509
13510
13511
13512  060142
13513  060142  012700  000200
13514  060146  170100
13515  060150  012700  060502
13516  060154  172410
13517  060156  012700  060512
13518  060162  172010
13519  060164  170205
13520  060166  170011
13521  060170  012700  060472
13522  060174  174010
13523  060176  012701  060502
13524  060202  012702  000004
13525  060206  022021
13526  060210  001401
13527  060212  104000
13528  060214  077204
```

```
        CLR     R0
        LDFPS   R0              ;ENTER FLOATING MODE.
        MOV     #WPAT00,R0
W16:    SUBF    (R0),AC0        ;TEST INSTRUCTION.
        STFPS   R5              ;GET FPS
        SETD                    ;RESET TO DOUBLE MODE
        MOV     #WPAT00,R0      ;GET THE RESULT.
        STD     AC0,(R0)
        MOV     #WPAT00,R1
        MOV     #4,R2
W17:    CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
        BEQ     W20
        EMT                     ;
W20:    SOB     R2,W17
        CMP     #4,R5           ;IS FPS CORRECT?
        BEQ     WDONE
        EMT                     ;

WPAT00: .WORD   0
WPAT01:         0
WPAT02:         0
WPAT03:         0

WDAP00: .WORD   0
WDAT01:         0
WDAT02:         0
WDAT03:         0

WDONE:
        JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
                                ;SEE IF THE USER HAS EXPRESSED
                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                ;THE USER TYPED CONTROL G?).

;*****************************************************************************************
;TEST 445       ADDD AND SUB WITH FSRC=0
;*****************************************************************************************
TS445:
        MOV     #200,R0
        LDFPS   R0              ;SET DOUBLE MODE
        MOV     #XPAT00,R0      ;SET AC0 TO POSITIVE
        LDD     (R0),AC0
        MOV     #XPAT10,R0      ;FSRC=0
X2:     ADDD    (R0),AC0        ;TEST INSTRUCTION
        STFPS   R5
        SETD
        MOV     #XDAT00,R0      ;GET RESULT.
        STD     AC0,(R0)
        MOV     #XPAT00,R1
        MOV     #4,R2
X3:     CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
        BEQ     X4
        EMT                     ;
X4:     SOB     R2,X3
```

```
13529 060216 012704 000200          MOV     #200,R4
13530 060222 020405                 CMP     R4,R5           ;IS FPS CORRECT?
13531 060224 001401                 BEQ     X5
13532 060226 104000                 EMT                     ;
13533 060230 012700 000200  X5:     MOV     #200,R0
13534 060234 170100                 LDFPS   R0              ;SET DOUBLE MODE
13535 060236 012700 060522          MOV     #XPAT20,R0      ;SET AC0 TO
13536 060242 172410                 LDD     (R0),AC0
13537 060244 012700 060512          MOV     #XPAT10,R0      ;FSRC=0
13538 060250 172010         X6:     ADDD    (R0),AC0        ;TEST INSTRUCTION
13539 060252 170205                 STFPS   R5
13540 060254 170011                 SETD
13541 060256 012700 060472          MOV     #XDAT00,R0      ;GET RESULT
13542 060262 174010                 STD     AC0,(R0)
13543 060264 012701 060522          MOV     #XPAT20,R1
13544 060270 012702 000004          MOV     #4,R2
13545 060274 022021         X7:     CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
13546 060276 001401                 BEQ     X10
13547 060300 104000                 EMT                     ;
13548 060302 077204         X10:    SOB     R2,X7
13549 060304 012704 000210          MOV     #210,R4
13550 060310 020405                 CMP     R4,R5           ;IS FPS CORRECT?
13551 060312 001401                 BEQ     X11
13552 060314 104000                 EMT                     ;
13553 060316 012700 000200  X11:    MOV     #200,R0
13554 060322 170100                 LDFPS   R0              ;SET DOUBLE MODE
13555 060324 012700 060502          MOV     #XPAT00,R0      ;SET AC0 TO NON-ZERO
13556 060330 172410                 LDD     (R0),AC0
13557 060332 012700 060512          MOV     #XPAT10,R0      ;FSRC=0
13558 060336 173010         X12:    SUBD    (R0),AC0        ;TEST INSTRUCTION
13559 060340 170205                 STFPS   R5
13560 060342 170011                 SETD
13561 060344 012700 060472          MOV     #XDAT00,R0      ;GET RESULT
13562 060350 174010                 STD     AC0,(R0)
13563 060352 012701 060502          MOV     #XPAT00,R1
13564 060356 012702 000004          MOV     #4,R2
13565 060362 022021         X13:    CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
13566 060364 001401                 BEQ     X14
13567 060366 104000                 EMT                     ;
13568 060370 077204         X14:    SOB     R2,X13
13569 060372 012704 000200          MOV     #200,R4         ;IS FPS CORRECT?
13570 060376 020405                 CMP     R4,R5
13571 060400 001401                 BEQ     X15
13572 060402 104000                 EMT                     ;
13573 060404 012700 000200  X15:    MOV     #200,R0
13574 060410 170100                 LDFPS   R0              ;SET DOUBLE MODE
13575 060412 012700 060522          MOV     #XPAT20,R0      ;SET AC0=A NEGATIVE
13576 060416 172410                 LDD     (R0),AC0
13577 060420 012700 060512          MOV     #XPAT10,R0      ;FSRC=0
13578 060424 173010         X16:    SUBD    (R0),AC0        ;TEST INSTRUCTION.
13579 060426 170205                 STFPS   R5
13580 060430 170011                 SETD
13581 060432 012700 060472          MOV     #XDAT00,R0      ;GET RESULT
13582 060436 174010                 STD     AC0,(R0)
13583 060440 012701 060522          MOV     #XPAT20,R1
13584 060444 012702 000004          MOV     #4,R2
```

```
13585  060450  022021                X17:    CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
13586  060452  001401                        BEQ     X20
13587  060454  104000                        EMT                     ;
13588  060456  077204                X20:    SOB     R2,X17
13589  060460  012704  000210              MOV     #210,R4         ;IS FPS CORRECT?
13590  060464  020405                        CMP     R4,R5
13591  060466  001421                        BEQ     XDONE
13592  060470  104000                        EMT                     ;
13593  060472  000000                XDAT00: .WORD   0
13594  060474  000000                XDAT01:         0
13595  060476  000000                XDAT02:         0
13596  060500  000000                XDAT03:         0
13597
13598  060502  010421                XPAT00: .WORD   010421
13599  060504  021042                XPAT01:         021042
13600  060506  031463                XPAT02:         031463
13601  060510  042104                XPAT03:         042104
13602
13603  060512  000000                XPAT10: .WORD   0
13604  060514  000000                XAPT11:         0
13605  060516  000000                XPAT12:         0
13606  060520  000000                XPAT13:         0
13607  060522  104210                XPAT20: .WORD   104210
13608  060524  114631                XPAT21:         114631
13609  060526  125252                XPAT22:         125252
13610  060530  135673                XPAT23:         135673
13611
13612  060532                        XDONE:
13613  060532  004767  037442                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
13614                                                                ;SEE IF THE USER HAS EXPRESSED
13615                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
13616                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
13617                                                                ;THE USER TYPED CONTROL G?).
13618
13619                                ;*****************************************************************************
13620                                ;TEST 446       SUBD WITH AC=0 TEST
13621                                ;*****************************************************************************
13622  060536                        TS446:
13623  060536  005037  060712                CLR     @#YFLAG
13624  060542  012737  060732  060714        MOV     #YPAT00,@#YTMP1 ;P
13625  060550  012737  060742  060716        MOV     #YPAT10,@#YTMP2 ;N
13626  060556  012737  000210  060720        MOV     #210,@#YTMP3
13627  060564  012700  000200        Y1:     MOV     #200,R0
13628  060570  170100                        LDFPS   R0              ;SET DOUBLE MODE
13629  060572  012700  060752                MOV     #YPAT20,R0      ;SET AC0=0
13630  060576  172410                        LDD     (R0),AC0
13631  060600  013700  060714                MOV     @#YTMP1,R0
13632  060604  173010                Y2:     SUBD    (R0),AC0        ;TEST INSTRUCTION
13633  060606  170205                        STFPS   R5
13634  060610  170011                        SETD
13635  060612  012700  060722                MOV     #YDAT00,R0      ;GET RESULT
13636  060616  174010                        STD     AC0,(R0)
13637  060620  012702  000004                MOV     #4,R2
13638  060624  013701  060716                MOV     @#YTMP2,R1      ;CHECK RESULT.
13639  060630  022021                Y3:     CMP     (R0)+,(R1)+
13640  060632  001401                        BEQ     1$
```

```
13641  060634  104000                          EMT                     ;
13642  060636  077204                  1$:     SOB     R2,Y3
13643  060640  023705  060720                  CMP     @#YTMP3,R5      ;FPS CORRECT?
13644  060644  001401                          BEQ     Y4
13645  060646  104000                          EMT                     ;
13646  060650  005737  060712          Y4:     TST     @#YFLAG         ;FINISHED TEST?
13647  060654  001015                          BNE     Y5
13648  060656  012737  177777  060712          MOV     #-1,@#YFLAG
13649  060664  012737  060742  060714          MOV     #YPAT10,@#YTMP1
13650  060672  012737  060732  060716          MOV     #YPAT00,@#YTMP2
13651  060700  012737  000200  060720          MOV     #200,@#YTMP3
13652  060706  000726                          BR      Y1
13653  060710  000424                  Y5:     BR      YDONE
13654
13655  060712  000000                  YFLAG:  .WORD   0
13656  060714  000000                  YTMP1:          0
13657  060716  000000                  YTMP2:          0
13658  060720  000000                  YTMP3:          0
13659
13660  060722  000000                  YDAT00: .WORD   0
13661  060724  000000                  YDAT01:         0
13662  060726  000000                  YDAT02:         0
13663  060730  000000                  YDAT03:         0
13664
13665  060732  063146                  YPAT00:         063146
13666  060734  052525                  YPAT01:         052525
13667  060736  042104                  YPAT02:         042104
13668  060740  167356                  YPAT03:         167356
13669
13670  060742  163146                  YPAT10:         163146
13671  060744  052525                  YPAT11:         052525
13672  060746  042104                  YPAT12:         042104
13673  060750  167356                  YPAT13:         167356
13674
13675  060752  000000                  YPAT20:         0
13676  060754  000000                  YPAT21:         0
13677  060756  000000                  YPAT22:         0
13678  060760  000000                  YPAT23:         0
13679
13680  060762                          YDONE:
13681  060762  004767  037212                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
13682                                                                  ;SEE IF THE USER HAS EXPRESSED
13683                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
13684                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
13685                                                                  ;THE USER TYPED CONTROL G?).
13686                          ;*****************************************************************************
13687                          ;TEST 447       ADDD WITH AC=0 TEST
13688                          ;*****************************************************************************
13689  060766                  TS447:
13690  060766  005067  000134                  CLR     ZFLAG
13691  060772  012737  061144  061130          MOV     #ZPAT00,@#ZTMP1 ;P
13692  061000  012737  000200  061132          MOV     #200,@#ZTMP2
13693  061006  012700  000200          Z1:     MOV     #200,R0
13694  061012  170100                          LDFPS   R0              ;SET DOUBLE MODE
13695  061014  012700  061164                  MOV     #ZPAT20,R0      ;SET AC0=0
13696  061020  172410                          LDD     (R0),AC0
```

```
13697  061022  013700  061130          MOV     @#ZTMP1,R0
13698  061026  172010          Z2:     ADDD    (R0),AC0        ;TEST INSTRUCTION
13699  061030  170205                  STFPS   R5
13700  061032  170011                  SETD
13701  061034  012700  061134          MOV     #ZDAT00,R0      ;GET RESULT
13702  061040  174010                  STD     AC0,(R0)
13703  061042  012702  000004          MOV     #4,R2
13704  061046  013701  061130          MOV     @#ZTMP1,R1      ;RESULT CORRECT?
13705  061052  022021          Z3:     CMP     (R0)+,(R1)+
13706  061054  001401                  BEQ     Z4
13707  061056  104000                  EMT                     ;
13708  061060  077204          Z4:     SOB     R2,Z3
13709  061062  023705  061132          CMP     @#ZTMP2,R5      ;FPS CORRECT?
13710  061066  001401                  BEQ     Z5
13711  061070  104000                  EMT                     ;
13712  061072  005737  061126          TST     @#ZFLAG         ;FINISHED TEST?
13713  061076  001012                  BNE     Z6
13714  061100  012737  177777  061126  MOV     #-1,@#ZFLAG
13715  061106  012737  061154  061130  MOV     #ZPAT10,@#ZTMP1
13716  061114  012737  000210  061132  MOV     #210,@#ZTMP2
13717  061122  000731                  BR      Z1
13718  061124  000423          Z6:     BR      ZDONE
13719
13720  061126  000000          ZFLAG:  .WORD   0
13721  061130  000000          ZTMP1:          0
13722  061132  000000          ZTMP2:          0
13723
13724  061134  000000          ZDAT00: .WORD   0
13725  061136  000000          ZDAT01:         0
13726  061140  000000          ZDAT02:         0
13727  061142  000000          ZDAT03:         0
13728
13729  061144  031463          ZPAT00:         031463
13730  061146  010421          ZPAT01:         010421
13731  061150  146314          ZPAT02:         146314
13732  061152  156735          ZPAT03:         156735
13733
13734  061154  156735          ZPAT10:         156735
13735  061156  167356          ZPAT11:         167356
13736  061160  135673          ZPAT12:         135673
13737  061162  146314          ZPAT13: .       146314
13738
13739  061164  000000          ZPAT20:         0
13740  061166  000000          ZPAT21:         0
13741  061170  000000          ZPAT22:         0
13742  061172  000000          ZPAT23:         0
13743
13744  061174                  ZDONE:
13745  061174  004767  037000          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
13746                                                          ;SEE IF THE USER HAS EXPRESSED
13747                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
13748                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
13749                                                          ;THE USER TYPED CONTROL G?).
13750
13751
13752                           ;*****************************************************************************
```

```
13753
13754
13755  061200
13756  061200  012700  003240      ;TEST 450       ADDF AND ADDD WITH E(AC)=E(FSRC) TEST AND (BUT FT) TEST
13757  061204  170100              ;*****************************************************************************
13758  061206  012700  061410      TS450:
13759                                      MOV     #3240,R0
13760  061212  172410                      LDFPS   R0              ;SET FIU FIV FD AND FT
13761  061214  012700  061420              MOV     #AAPAT0,R0      ;FLOWS IN TRAP WILL
13762  061220  172010                                              ;OCCUR
13763                                      LDD     (R0),AC0        ;SET UP AC0
13764  061222  012700  061400              MOV     #AAPAT1,R0
13765  061226  174010              AA2:    ADDD    (R0),AC0        ;TEST INSTRUCTION
13766  061230  012701  061430                                      ;SHOULD TRUNCATE
13767  061234  012702  000004      AA3:    MOV     #AADAT0,R0
13768  061240  022021                      STD     AC0,(R0)        ;GET THE RESULT
13769  061242  001401                      MOV     #AAPAT2,R1
13770  061244  104000                      MOV     #4,R2
13771  061246  077204              AA4:    CMP     (R0)+,(R1)+     ;CORRECT?
13772                                      BEQ     AA7
13773                                      EMT                     ;
13774                               AA7:   SOB     R2,AA4
13775
13776  061250  012700  003200      ;NOW TEST DOUBLE FLOATING ROUND MODE.
13777  061254  170100              ;A 1 SHOULD BE ADDED TO THE LSB ON ROUNG MODE.
13778  061256  012700  061410
13779  061262  172410                      MOV     #3200,R0        ;SET FD FIV FIV. FT=0
13780  061264  012700  061420              LDFPS   R0
13781  061270  172010                      MOV     #AAPAT0,R0
13782                                      LDD     (R0),AC0        ;SET UP AC0 OPERAND
13783  061272  012700  061400              MOV     #AAPAT1,R0
13784  061276  174010              AA11:   ADDD    (R0),AC0        ;TEST INSTRUCTION
13785  061300  012701  061440                                      ;SHOULD ROUND
13786  061304  012702  000004      AA12:   MOV     #AADAT0,R0
13787  061310  022021                      STD     AC0,(R0)        ;GET THE RESULT
13788  061312  001401                      MOV     #AAPAT3,R1
13789  061314  104000                      MOV     #4,R2
13790  061316  077204              AA13:   CMP     (R0)+,(R1)+     ;CORRECT?
13791                                      BEQ     AA20
13792                                      EMT                     ;
13793                               AA20:  SOB     R2,AA13
13794  061320  012700  003200
13795  061324  170100              ;NOW TEST ADDF WITH FT=0, ROUND MODE
13796  061326  012700  061410
13797  061332  172410                      MOV     #3200,R0                ;FIV=1, FIV=1, FT=0
13798  061334  170001                      LDFPS   R0
13799  061336  012700  061460              MOV     #AAPAT0,R0      ;LOAD AC0 OPERAND
13800  061342  172010                      LDD     (R0),AC0
13801                                      SETF                    ;ENTER FLOATING MODE
13802  061344                              MOV     #AAPAT5,R0
13803  061344  170011              AA22:   ADDF    (R0),AC0        ;TEST INSTRUCTION
13804                                                              ;SHOULD ROUND
13805  061346  012700  061400      AA23:
13806  061352  174010                      SETD                    ;RESET TO DOUBLE
13807  061354  012701  061470                                      ;MODE
13808  061360  012702  000002              MOV     #AADAT0,R0      ;GET THE RESULT
                                     STD     AC0,(R0)
                                     MOV     #AAPAT6,R1      ;CORRECT?
                                     MOV     #2,R2
```

```
13809  061364  022021
13810  061366  001401
13811  061370  104000
13812  061372  077204
13813  061374  000137  061500
13814  061400  000000
13815  061402  000000
13816  061404  000000
13817  061406  000000
13818  061410  000200
13819  061412  000000
13820  061414  000000
13821  061416  000000
13822  061420  000200
13823  061422  000000
13824  061424  000000
13825  061426  000001
13826  061430  000400
13827  061432  000000
13828  061434  000000
13829  061436  000000
13830  061440  000400
13831  061442  000000
13832  061444  000000
13833  061446  000001
13834  061450  000400
13835  061452  000000
13836  061454  100000
13837  061456  000000
13838  061460  000200
13839  061462  000001
13840  061464  000000
13841  061466  000000
13842  061470  000400
13843  061472  000001
13844  061474  000000
13845  061476  000000
13846  061500
13847  061500  004767  036474
13848
13849
13850
13851
13852
13853
13854
13855  061504
13856
13857  061504  012704  003200
13858  061510  170104
13859  061512  012700  062160
13860  061516  172410
13861  061520  012700  062200
13862  061524  172010
13863  061526  170205
13864  061530  012700  062150
```

```
AA24:   CMP     (R0)+,(R1)+
        BEQ     AA27
        EMT                             ;
AA27:   SOB     R2,AA24
        JMP     @#AADONE
AADAT0:         0
                0
                0
                0
AAPAT0:         200
                0
                0
                0
AAPAT1:         200
                0
                0
                1
AAPAT2:         400
                0
                0
                0
AAPAT3:         400
                0
                0
                1
AAPAT4:         400
                0
                100000
                0
AAPAT5:         200
                1
                0
                0
AAPAT6:         400
                1
                0
                0
AADONE:
        JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                ;SEE IF THE USER HAS EXPRESSED
                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                ;THE USER TYPED CONTROL G?).
;*******************************************************************************
;TEST 451       ADDF AND ADDD WITH E(AC) LESS THAN E(FSRC) TEST
;*******************************************************************************
TS451:
;EXPONENT DIFFERENCE=57=71 (OCT) FD=1
        MOV     #3200,R4        ;SET FIV,FIV, AND FD
        LDFPS   R4
        MOV     #CCP0,R0        ;SET AC0 OPERAND
        LDD     (R0),AC0                ;AC0
        MOV     #CCP2,R0
CCX2:   ADDD    (R0),AC0        ;TEST INSTRUCTION
        STFPS   R5              ;GET FPS
        MOV     #CCDAT0,R0      ;GET THE RESULT
```

```
13865  061534  174010                  STD     AC0,(R0)
13866  061536  012701  062200          MOV     #CCP2,R1        ;IS IT CORRECT
13867  061542  012702  000004          MOV     #4,R2
13868  061546  022021           CCX3:  CMP     (R0)+,(R1)+
13869  061550  001401                  BEQ     CCX6
13870  061552  104000                  EMT                     ;
13871  061554  077204           CCX6:  SOB     R2,CCX3
13872  061556  020405                  CMP     R4,R5           ;FPS CORRECT?
13873  061560  001401                  BEQ     CCX7
13874  061562  104000                  EMT                     ;
13875                          ;EXPONENT DIFFERENCE=56=70 (OCT) FD=1
13876  061564  012704  003200  CCX7:  MOV     #3200,R4        ;SET FIV,FIV, AND FD
13877  061570  170104                  LDFPS   R4
13878  061572  012700  062160          MOV     #CCP0,R0        ;SET AC0 OPERAND
13879  061576  172410                  LDD     (R0),AC0
13880  061600  012700  062170          MOV     #CCP1,R0        ;FSRC
13881  061604  172010           CCX8:  ADDD    (R0),AC0        ;TEST INSTRUCTION
13882  061606  170205                  STFPS   R5              ;GET FPS
13883  061610  012700  062150          MOV     #CCDAT0,R0      ;GET THE RESULT
13884  061614  174010                  STD     AC0,(R0)
13885  061616  012701  062250          MOV     #CCP7,R1        ;IS IT CORRECT
13886  061622  012702  000004          MOV     #4,R2
13887  061626  022021           CCX9:  CMP     (R0)+,(R1)+
13888  061630  001401                  BEQ     CCX12
13889  061632  104000                  EMT                     ;
13890  061634  077204           CCX12: SOB     R2,CCX9
13891  061636  020405                  CMP     R4,R5           ;FPS CORRECT?
13892  061640  001401                  BEQ     CCX13
13893  061642  104000                  EMT                     ;
13894                          ;EXPONENT DIFFERENCE=25=31 (OCT) FD=0
13895  061644  012700  062160  CCX13: MOV     #CCP0,R0        ;SET UP AC0 OPERAND.
13896  061650  172410                  LDD     (R0),AC0
13897  061652  012704  003000          MOV     #3000,R4        ;SET FIV,FIV. CLEAR FD.
13898  061656  170104                  LDFPS   R4
13899  061660  012700  062240          MOV     #CCP6,R0        ;FSRC
13900  061664  172010           CCX14: ADDF    (R0),AC0        ;TEST INSTRUCTION
13901  061666  170205                  STFPS   R5
13902  061670  170011                  SETD                    ;REENTER DOUBLE MOVE
13903  061672  012700  062150          MOV     #CCDAT0,R0      ;GET THE RESULT
13904  061676  174010                  STD     AC0,(R0)
13905  061700  012701  062240          MOV     #CCP6,R1        ;IS THE RESULT CORRECT?
13906  061704  012702  000002          MOV     #2,R2
13907  061710  022021           CCX15: CMP     (R0)+,(R1)+
13908  061712  001401                  BEQ     CCX18
13909  061714  104000                  EMT                     ;
13910  061716  077204           CCX18: SOB     R2,CCX15
13911  061720  020405                  CMP     R4,R5
13912  061722  001401                  BEQ     CCX19
13913  061724  104000                  EMT                     ;
13914                          ;EXPONENT DIFFERENCE=24=30 (OCT) FD=0
13915  061726  012700  062210  CCX19: MOV     #CCP3,R0        ;SET UP AC0 OPERAND.
13916  061732  172410                  LDD     (R0),AC0
13917  061734  012704  003000          MOV     #3000,R4        ;SET FIV,FIV. CLEAR FD.
13918  061740  170104                  LDFPS   R4
13919  061742  012700  062230          MOV     #CCP5,R0        ;FSRC
13920  061746  172010           CCX20: ADDF    (R0),AC0        ;TEST INSTRUCTION
```

```
13921  061750  170205                STFPS   R5
13922  061752  170011                SETD                    ;REENTER DOUBLE MOVE
13923  061754  012700  062150        MOV     #CCDATO,R0      ;GET THE RESULT
13924  061760  174010                STD     AC0,(R0)
13925  061762  012701  062260        MOV     #CCP10,R1       ;IS THE RESLT CORRECT?
13926  061766  012702  000002        MOV     #2,R2
13927  061772  022021        CCX21:  CMP     (R0)+,(R1)+
13928  061774  001401                BEQ     CCX24
13929  061776  104000                EMT                     ;
13930  062000  077204        CCX24:  SOB     R2,CCX21
13931  062002  020405                CMP     R4,R5
13932  062004  001401                BEQ     CCX25
13933  062006  104000                EMT                     ;
13934                        ;EXPONENT DIFFERENCE=1 FD=1
13935  062010  012704  003200        CCX25:  MOV     #3200,R4        ;SET FIV,FIV, AND FD
13936  062014  170104                LDFPS   R4
13937  062016  012700  062160        MOV     #CCP0,R0        ;SET AC0 OPERAND
13938  062022  172410                LDD     (R0),AC0
13939  062024  012700  062210        MOV     #CCP3,R0        ;FSRC
13940  062030  172010        CCX26:  ADDD    (R0),AC0        ;TEST INSTRUCTION
13941  062032  170205                STFPS   R5              ;GET FPS
13942  062034  012700  062150        MOV     #CCDATO,R0      ;GET THE RESULT
13943  062040  174010                STD     AC0,(R0)
13944  062042  012701  062270        MOV     #CCP11,R1       ;IS IT CORRECT
13945  062046  012702  000004        MOV     #4,R2
13946  062052  022021        CCX27:  CMP     (R0)+,(R1)+
13947  062054  001401                BEQ     CCX30
13948  062056  104000                EMT                     ;
13949  062060  077204        CCX30:  SOB     R2,CCX27
13950  062062  020405                CMP     R4,R5           ;FPS CORRECT?
13951  062064  001401                BEQ     CCX31
13952  062066  104000                EMT                     ;
13953                        ;EXPONENT DIFFERENCE=100=144 (OCT) FD=1
13954  062070  012704  003200        CCX31:  MOV     #3200,R4        ;SE, FIV,FIV, AND FD
13955  062074  170104                LDFPS   R4
13956  062076  012700  062160        MOV     #CCP0,R0        ;SET AC0 OPERAND
13957  062102  172410                LDD     (R0),AC0
13958  062104  012700  062220        MOV     #CCP4,R0        ;FSRC
13959  062110  172010        CCX32:  ADDD    (R0),AC0        ;TEST INSTRUCTION
13960  062112  170205                STFPS   R5              ;GET FPS
13961  062114  012700  062150        MOV     #CCDATO,R0      ;GET THE RESULT
13962  062120  174010                STD     AC0,(R0)
13963  062122  012701  062220        MOV     #CCP4,R1        ;IS IT CORRECT
13964  062126  012702  000004        MOV     #4,R2
13965  062132  022021        CCX33:  CMP     (R0)+,(R1)+
13966  062134  001401                BEQ     CCX36
13967  062136  104000                EMT                     ;
13968  062140  077204        CCX36:  SOB     R2,CCX33
13969  062142  020405                CMP     R4,R5           ;FPS CORRECT?
13970  062144  001461                BEQ     CCXDONE
13971  062146  104000                EMT                     ;
13972  062150  000000        CCDATO: 0
13973  062152  000000                0
13974  062154  000000                0
13975  062156  000000                0
13976  062160  000200        CCP0:   200                     ;E(AC)=1
```

```
13977  062162  000000                    0
13978  062164  000000                    0
13979  062166  000000                    0
13980  062170  016200            CCP1:   16200                   ;E(FSRC)=E(AC)+56=57
13981  062172  000000                    0                       ;                =71(OCT)
13982  062174  000000                    0
13983  062176  000000                    0
13984  062200  016400            CCP2:   16400                   ;E(FSRC)=E(AC)+57=58
13985  062202  000000                    0                       ;                =72(OCT)
13986  062204  000000                    0
13987  062206  000000                    0
13988  062210  000400            CCP3:   400                     ;E(FSRC)=E(AC)+1=2
13989  062212  000000                    0
13990  062214  000000                    0
13991  062216  000000                    0
13992  062220  031200            CCP4:   31200                   ;E(FSRC)=E(AC)+100=101=145(OCT)
13993  062222  000000                    0
13994  062224  000000                    0
13995  062226  000000                    0
13996  062230  006200            CCP5:   6200                    ;E(FSRC)=E(AC)+24=25=31(OCT)
13997  062232  000000                    0
13998  062234  000000                    0
13999  062236  000000                    0
14000  062240  006400            CCP6:   6400                    ;E(FSRC)=E(AC)+25=26=32(OCT)
14001  062242  000000                    0
14002  062244  000000                    0
14003  062246  000000                    0
14004  062250  016200            CCP7:   16200                   ;CCP1 RES
14005  062252  000000                    0
14006  062254  000000                    0
14007  062256  000001                    1
14008  062260  006200            CCP10:  6200                    ;CCP5 RES
14009  062262  000001                    1
14010  062264  000000                    0
14011  062266  000000                    0
14012  062270  000500            CCP11:  500                     ;CCP3 RES
14013  062272  000000                    0
14014  062274  000000                    0
14015  062276  000000                    0
14016  062300  000200            CCP12:  200                     ;BAD CONSTANT
14017  062302  000000                    0                       ;RES CCP2,CCP4
14018  062304  000000                    0
14019  062306  000000                    0
14020
14021  062310                    CCXDONE:
14022  062310  004767  035664              JSR     PC,..RSET     ;GO INITIALIZE THE FPS AND STACK; AND
14023                                                            ;SEE IF THE USER HAS EXPRESSED
14024                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
14025                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
14026                                                            ;THE USER TYPED CONTROL G?).
14027
14028                             ;*****************************************************************************
14029                             ;TEST 452       ADDF AND ADDD WITH E(AC) GREATER THAN E(FSRC) TEST
14030                             ;*****************************************************************************
14031  062314                     TS452:
14032                             ;EXPONENT DIFFERENCE=57=71 (OCT) FD=1
```

```
14033  062314  012704  003200          MOV     #3200,R4        ;SET FIV FIV, AND FD
14034  062320  170104                  LDFPS   R4
14035  062322  012700  063010          MOV     #BBPAT2,R0      ;SET AC0 OPERAND.
14036  062326  172410                  LDD     (R0),AC0
14037  062330  012700  063000          MOV     #BBPAT1,R0      ;FSRC
14038  062334  172010          BB2:    ADDD    (R0),AC0        ;TEST INSTRUCTION
14039  062336  170205                  STFPS   R5
14040  062340  012700  062760  BB3:    MOV     #BBDAT0,R0      ;GET THE RESULT
14041  062344  174010                  STD     AC0,(R0)
14042  062346  012701  063010          MOV     #B3PAT2,R1      ;RESULT CORRECT?
14043  062352  012702  000004          MOV     #4,R2
14044  062356  022021          BB4:    CMP     (R0)+,(R1)+
14045  062360  001401                  BEQ     BB5
14046  062362  104000                  EMT                     ;
14047  062364  077204          BB5:    SOB     R2,BB4
14048                                                          ;WAS FPS CORRECT?
14049  062366  020405                  CMP     R4,R5
14050  062370  001401                  BEQ     BB6
14051  062372  104000                  EMT                     ;
14052          ;EXPONENT DIFFERENCE=56=70 (OCT) FD=1
14053  062374  012704  003200  BB6:    MOV     #3200,R4        ;SET FIV,FIV, AND FD
14054  062400  170104                  LDFPS   R4
14055  062402  012700  063030          MOV     #BBPAT4,R0      ;SET AC0 OPERAND
14056  062406  172410                  LDD     (R0),AC0
14057  062410  012700  063000          MOV     #BBPAT1,R0      ;FSRC
14058  062414  172010          BB7:    ADDD    (R0),AC0        ;TEST INSTRUCTION
14059  062416  170205                  STFPS   R5              ;GET FPS
14060  062420  012700  062760          MOV     #BBDAT0,R0      ;GET THE RESULT
14061  062424  174010                  STD     AC0,(R0)
14062  062426  012701  063070          MOV     #BBP10,R1       ;IS IT CORRECT
14063  062432  012702  000004          MOV     #4,R2
14064  062436  022021          BB10:   CMP     (R0)+,(R1)+
14065  062440  001401                  BEQ     BB13
14066  062442  104000                  EMT                     ;
14067  062444  077204          BB13:   SOB     R2,BB10
14068  062446  020405                  CMP     R4,R5           ;FPS CORRECT?
14069  062450  001401                  BEQ     BB14
14070  062452  104000                  EMT                     ;
14071          ;EXPONENT DIFFERENCE=25=31 (OCT) FD=0
14072  062454  012700  062770  BB14:   MOV     #BBPAT0,R0      ;SET UP AC0 OPERAND
14073  062460  172410                  LDD     (R0),AC0
14074  062462  012704  003000          MOV     #3000,R4        ;SET FIV AND FIV
14075                                                          ;CLEAR FD
14076  062466  170104                  LDFPS   R4
14077  062470  012700  063000          MOV     #BBPAT1,R0      ;FSRC
14078  062474  172010          BB15:   ADDF    (R0),AC0        ;TEST INSTRUCTION
14079  062476  170205                  STFPS   R5
14080  062500  170011                  SETD                    ;REENTED DOUBLE MODE.
14081  062502  012700  062760          MOV     #BBDAT0,R0      ;GET THE RESULT
14082  062506  174010                  STD     AC0,(R0)
14083  062510  012701  062770          MOV     #BBPAT0,R1      ;IS THE RESULT
14084  062514  012702  000002          MOV     #2,R2           ;CORRECT?
14085  062520  022021          BB16:   CMP     (R0)+,(R1)+
14086  062522  001401                  BEQ     BB17
14087  062524  104000                  EMT                     ;
14088  062526  077204          BB17:   SOB     R2,BB16
```

```
14089  062530  020405                        CMP     R4,R5           ;IS FPS CORRECT?
14090  062532  001401                        BEQ     BB20
14091  062534  104000                        EMT                     ;
14092                                ;EXPONENT DIFFERENCE=24=30 (OCT)
14093  062536  012700  063020        BB20:   MOV     #BBPAT3,R0      ;SET UP AC0 OPERAND.
14094  062542  172410                        LDD     (R0),AC0
14095  062544  012704  003000                MOV     #3000,R4        ;SET FIU,FIV. CLEAR FD.
14096  062550  170104                        LDFPS   R4
14097  062552  012700  063000                MOV     #BBPAT1,R0      ;FSRC
14098  062556  172010        BB21:   ADDF    (R0),AC0        ;TEST INSTRUCTION
14099  062560  170205                        STFPS   R5
14100  062562  170011                        SETD                    ;REENTER DOUBLE MODE
14101  062564  012700  062760                MOV     #BBDAT0,R0      ;GET THE RESULT
14102  062570  174010                        STD     AC0,(R0)
14103  062572  012701  063060                MOV     #BBP7,R1        ;IS THE RESULT CORRECT?
14104  062576  012702  000002                MOV     #2,R2
14105  062602  022021        BB22:   CMP     (R0)+,(R1)+
14106  062604  001401                        BEQ     BB25
14107  062606  104000                        EMT                     ;
14108  062610  077204        BB25:   SOB     R2,BB22
14109  062612  020405                        CMP     R4,R5
14110  062614  001401                        BEQ     BB26
14111  062616  104000                        EMT                     ;
14112                                ;EXPONENT DIFFERENCE=1
14113  062620  012704  003200        BB26:   MOV     #3200,R4
14114  062624  170104                        LDFPS   R4              ;SET UP AC0 OPERAND
14115  062626  012700  063040                MOV     #BBPAT5,R0
14116  062632  172410                        LDD     (R0),AC0
14117  062634  012700  063000                MOV     #BBPAT1,R0      ;FSRC
14118  062640  172010        BB27:   ADDD    (R0),AC0        ;TEST INSTRUCTION
14119  062642  170205                        STFPS   R5
14120  062644  012700  062760                MOV     #BBDAT0,R0      ;GET THE RESULT.
14121  062650  174010                        STD     AC0,(R0)
14122  062652  012701  063100                MOV     #BBP11,R1       ;IS IT CORRECT?
14123  062656  012702  000004                MOV     #4,R2
14124  062662  022021        BB30:   CMP     (R0)+,(R1)+
14125  062664  001401                        BEQ     BB31
14126  062666  104000                        EMT                     ;
14127  062670  077204        BB31:   SOB     R2,BB30
14128  062672  020405                        CMP     R4,R5           ;IS FPS CORRECT
14129  062674  001401                        BEQ     BB32
14130  062676  104000                        EMT                     ;
14131                                ;EXPONENT DIFFERENCE=100=144 (OCT)
14132  062700  012704  003200        BB32:   MOV     #3200,R4
14133  062704  170104                        LDFPS   R4              ;SET FIV,FIV AND FD
14134  062706  012700  063050                MOV     #BBPAT6,R0      ;SET UP AC0 OPERAND.
14135  062712  172410                        LDD     (R0),AC0
14136  062714  012700  063000                MOV     #BBPAT1,R0      ;FSRC
14137  062720  172010        BB33:   ADDD    (R0),AC0        ;TEST INSTRUCTION
14138  062722  170205                        STFPS   R5
14139  062724  012700  062760                MOV     #BBDAT0,R0      ;GET THE RESULT
14140  062730  174010                        STD     AC0,(R0)
14141  062732  012701  063050                MOV     #BBPAT6,R1      ;IS IT CORRECT
14142  062736  012702  000004                MOV     #4,R2
14143  062742  022021        BB34:   CMP     (R0)+,(R1)+
14144  062744  001401                        BEQ     BB35
```

```
14145  062746  104000                    EMT                     ;
14146  062750  077204          BB35:     SOB     R2,BB34
14147  062752  020405                    CMP     R4,R5           ;IS FPS CORRECT
14148  062754  001455                    BEQ     BBDONE
14149  062756  104000                    EMT                     ;
14150  062760  000000          BBDAT0:   0
14151  062762  000000                    0
14152  062764  000000                    0
14153  062766  000000                    0
14154  062770  006400          BBPAT0:   6400                    ;F(AC)=E(FSRC)+25=26
14155  062772  000000                    0                       ;                 =32(OCT)
14156  062774  000000                    0
14157  062776  000000                    0
14158  063000  000200          BBPAT1:   200
14159  063002  000000                    0                       ;E(FSRC)=1
14160  063004  000000                    0
14161  063006  000000                    0
14162  063010  016400          BBPAT2:   16400
14163  063012  000000                    0                       ;E(AC)=E(FSRC)+57=58
14164  063014  000000                    0                       ;                 =72(OCT)
14165  063016  000000                    0
14166  063020  006200          BBPAT3:   6200                    ;E(AC)=E(FSRC)+24=25
14167  063022  000000                    0                       ;                 =31(OCT)
14168  063024  000000                    0
14169  063026  000000                    0
14170  063030  016200          BBPAT4:   16200                   ;E(AC)=E(FSR6)+56=57
14171  063032  000000                    0                       ;                 =71(OCT)
14172  063034  000000                    0
14173  063036  000000                    0
14174  063040  000400          BBPAT5:   400                     ;E(AC)=E(FSRC)+1=2
14175  063042  000000                    0
14176  063044  000000                    0
14177  063046  000000                    0
14178  063050  031200          BBPAT6:   31200                   ;E(AC)=E(FSRC)+100=101
14179  063052  000000                    0                       ;                  =145(OCT)
14180  063054  000000                    0
14181  063056  000000                    0
14182  063060  006200          BBP7:     6200                    ;BBPAT3 RES
14183  063062  000001                    1
14184  063064  000000                    0
14185  063066  000000                    0
14186  063070  016200          BBP10:    16200                   ;BBPAT4 RES
14187  063072  000000                    0
14188  063074  000000                    0
14189  063076  000001                    1
14190  063100  000500          BBP11:    500                     ;BBPAT5 RES
14191  063102  000000                    0
14192  063104  000000                    0
14193  063106  000000                    0
14194  063110                  BBDONE:
14195  063110  004767  035064            JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
14196                                                            ;SEE IF THE USER HAS EXPRESSED
14197                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
14198                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
14199                                                            ;THE USER TYPED CONTROL G?).
14200                                  ;*************************************************************************************
```

```
14201
14202
14203  063114
14204
14205  063114  012704  003200
14206  063120  170104
14207  063122  012700  063700
14208  063126  172410
14209  063130  012700  063700
14210  063134  172010
14211  063136  170205
14212  063140  012700  063660
14213  063144  174010
14214  063146  012701  064000
14215  063152  012702  000004
14216  063156  022021
14217  063160  001401
14218  063162  104000
14219  063164  077204
14220  063166  052704  000010
14221  063172  020405
14222  063174  001401
14223  063176  104000
14224
14225  063200  012704  003200
14226  063204  170104
14227  063206  012700  063710
14228  063212  172410
14229  063214  012700  063700
14230  063220  172010
14231  063222  170205
14232  063224  012700  063660
14233  063230  174010
14234  063232  012701  063670
14235  063236  012702  000004
14236  063242  022021
14237  063244  001401
14238  063246  104000
14239  063250  077204
14240  063252  052704  000004
14241  063256  020405
14242  063260  001401
14243  063262  104000
14244
14245  063264  012704  003200
14246  063270  170104
14247  063272  012700  063700
14248  063276  172410
14249  063300  012700  063710
14250  063304  172010
14251  063306  170205
14252  063310  012700  063660
14253  063314  174010
14254  063316  012701  063670
14255  063322  012702  000004
14256  063326  022021
```

```
;TEST 453       ADDD WITH NEGATVE OPRANDS TEST
;*************************************************************************************
;
TS453:
;BOTH OPERANDS NEGATIVE
        MOV     #3200,R4        ;SET FIO, FIV, AND FD
        LDFPS   R4
        MOV     #DDP1,R0                ;SET AC0 OPERAND
        LDD     (R0),AC0
        MOV     #DDP1,R0                ;ESRC
DD2:    ADDD    (R0),AC0                ;TEST INSTRUCTION
        STFPS   R5                      ;GET FPS
        MOV     #DDDAT0,R0              ;GET THE RESULT
        STD     AC0,(R0)
        MOV     #DDP9,R1                ;IS IT CORRECT
        MOV     #4,R2
DD3:    CMP     (R0)+,(R1)+
        BEQ     DD6
        EMT                     ;
DD6:    SOB     R2,DD3
        BIS     #10,R4
        CMP     R4,R5                   ;FPS CORRECT?
        BEQ     DD7
        EMT                     ;
;AC POS FSRC NEG         AC=-FSRC
DD7:    MOV     #3200,R4                ;SET FIO, FIV, AND FD
        LDFPS   R4
        MOV     #DDP2,R0                ;SET AC0 OPERAND
        LDD     (R0),AC0
        MOV     #DDP1,R0                ;FSPC
DD8:    ADDD    (R0),AC0                ;TEST INSTRUCTION
        STFPS   R5                      ;GET FPS
        MOV     #DDDAT0,R0              ;GET THE RESULT
        STD     AC0,(R0)
        MOV     #DDP0,R1                ;IS IT CORRECT
        MOV     #4,R2
DD10:   CMP     (R0)+,(R1)+
        BEQ     DD11
        EMT             ;
DD11:   SOB     R2,DD10
        BIS     #4,R4
        CMP     R4,R5                   ;FPS CORRECT?
        BEQ     DD12
        EMT             ;
;AC NEG FSRC POS         AC=-FSRC
DD12:   MOV     #3200,R4                ;SET FIU, FIV, AND FD
        LDFPS   R4
        MOV     #DDP1,R0                ;SET AC0 OPERAND
        LDD     (R0),AC0
        MOV     #DDP2,R0                ;FSRC
DD13:   ADDD    (R0),AC0                ;TEST INSTRUCTION
        STFPS   R5                      ;GET FPS
        MOV     #DDDAT0,R0              ;GET THE RESULT
        STD     AC0,(R0)
        MOV     #DDP0,R1                ;IS IT CORRECT
        MOV     #4,R2
DD14:   CMP     (R0)+,(R1)+
```

```
14257  063330  001401                BEQ     DD15
14258  063332  104000                EMT                     ;
14259  063334  077204        DD15:   SOB     R2,DD14
14260  063336  052704  000004        BIS     #4,R4
14261  063342  020405                CMP     R4,R5                   ;EPS CORRECT?
14262  063344  001401                BEQ     DD16
14263  063346  104000                EMT                     ;
14264                        ;AC0 POC        FSRC    NEG     /AC/ > /FSRC/
14265  063350  012704  003200        DD16:   MOV     #3200,R4                ;SET FIV, FIV AND FD
14266  063354  170104                LDFPS   R4
14267  063356  012700  063720        MOV     #DDP3,R0                ;SET AC0 OPERAND
14268  063362  172410                LDD     (R0),AC0
14269  063364  012700  063750        MOV     #DDP6,R0                ;ESPC
14270  063370  172010        DD17:   ADDD    (R0),AC0                ;TEST INSTRUCTION
14271  063372  170205                STFPS   R5                      ;GET FPS
14272  063374  012700  063660        MOV     #DDDAT0,R0              ;GET THE RESULT
14273  063400  174010                STD     AC0,(R0)
14274  063402  012701  063760        MOV     #DDP7,R1                ;IS IT CORRECT
14275  063406  012702  000004        MOV     #4,R2
14276  063412  022021        DD18:   CMP     (R0)+,(R1)+
14277  063414  001401                BEQ     DD21
14278  063416  104000                EMT                     ;
14279  063420  077204        DD21:   SOB     R2,DD18
14280  063422  020405                CMP     R4,R5                   ;EPS CORRECT?
14281  063424  001401                BEQ     DD22
14282  063426  104000                EMT                     ;
14283                        ;AC NEG FSRC    POS     /FSRC/ > /AC/
14284  063430  012704  003200        DD22:   MOV     #3200,R4                ;SET FIO,FIV, AND FD
14285  063434  170104                LDFPS   R4
14286  063436  012700  063750        MOV     #DDP6,R0                ;SET AC0 OPERAND
14287  063442  172410                LDD     (R0),AC0
14288  063444  012700  063720        MOV     #DDP3,R0                ;FSPC
14289  063450  172010        DD23:   ADDD    (R0),AC0                ;TEST INSTRUCTION
14290  063452  170205                STFPS   R5                      ;GET FPS
14291  063454  012700  063660        MOV     #DDDAT0,R0              ;GET THE RESULT
14292  063460  174010                STD     AC0,(R0)
14293  063462  012701  063760        MOV     #DDP7,R1                ;IS IT CORRECT?
14294  063466  012702  000004        MOV     #4,R2
14295  063472  022021        DD24:   CMP     (R0)+,(R1)+
14296  063474  001401                BEQ     DD27
14297  063476  104000                EMT                     ;
14298  063500  077204        DD27:   SOB     R2,DD24
14299  063502  020405                CMP     R4,R5                   ;FPS CORRECT?
14300  063504  001401                BEQ     DD30
14301  063506  104000                EMT                     ;
14302                        ;AC0 POS        FSRC    NEG     /AC/</FRSRC/
14303  063510  012704  003200        DD30:   MOV     #3200,R4                ;SET FIO,FIV,AND FD
14304  063514  170104                LDFPS   R4
14305  063516  012700  063730        MOV     #DDP4,R0                ;SET AC0 OPERAND
14306  063522  172410                LDD     (R0),AC0
14307  063524  012700  063740        MOV     #DDP5,R0                ;FSPC
14308  063530  172010        DD31:   ADDD    (R0),AC0                ;TEST INSTRUCTION
14309  063532  170205                STFPS   R5                      ;GET FPS
14310  063534  012700  063660        MOV     #DDDAT0,R0              ;GET THE RESULT
14311  063540  174010                STD     AC0,(R0)
14312  063542  012701  063770        MOV     #DDP8,R1                ;IS IT CORRECT
```

```
14313  063546  012702  000004          MOV     #4,R2
14314  063552  022021          DD32:   CMP     (R0)+,(R1)+
14315  063554  001401                  BEQ     DD35
14316  063556  104000                  EMT                             ;
14317  063560  077204          DD35:   SOB     R2,DD32
14318  063562  052704  000010          BIS     #10,R4
14319  063566  020405                  CMP     R4,R5                   ;FPS CORRECT?
14320  063570  001401                  BEQ     DD36
14321  063572  104000                  EMT                             ;
14322                          ;ACO NEG        FSRC    POS     /FSRC/</AC/
14323  063574  012704  003200  DD36:   MOV     #3200,R4                ;SET FIO, FIV, AND FD
14324  063600  170104                  LDFPS   R4
14325  063602  012700  063740          MOV     #DDP5,R0                ;SET ACO OPERAND
14326  063606  172410                  LDD     (R0),AC0
14327  063610  012700  063730          MOV     #DDP4,R0                ;FSPC
14328  063614  172010          DD37:   ADDD    (R0),AC0                ;TEST INSTRUCTION
14329  063616  170205                  STFPS   R5                      ;GET FPS
14330  063620  012700  063660          MOV     #DDDATO,R0              ;GET THE RESULT
14331  063624  174010                  STD     AC0,(R0)
14332  063626  012701  063770          MOV     #DDP8,R1                ;IS IT CORRECT
14333  063632  012702  000004          MOV     #4,R2
14334  063636  022021          DD38:   CMP     (R0)+,(R1)+
14335  063640  001401                  BEQ     DD41
14336  063642  104000                  EMT                             ;
14337  063644  077204          DD41:   SOB     R2,DD38
14338  063646  052704  000010          BIS     #10,R4
14339  063652  020405                  CMP     R4,R5                   ;FPS CORRECT?
14340  063654  001455                  BEQ     DDDONE
14341  063656  104000                  EMT                             ;
14342  063660  000000          DDDATO: 0
14343  063662  000000                  0
14344  063664  000000                  0
14345  063666  000000                  0
14346  063670  000000          DDP0:   0
14347  063672  000000                  0
14348  063674  000000                  0
14349  063676  000000                  0
14350  063700  100200          DDP1:   100200                          ;-DDP2
14351  063702  000000                  0
14352  063704  000000                  0
14353  063706  000000                  0
14354  063710  000200          DDP2:   200                             ;-DDP1
14355  063712  000000                  0
14356  063714  000000                  0
14357  063716  000000                  0
14358  063720  001100          DDP3:   1100                            ;EXP=4
14359  063722  000000                  0                               ;FRAC=...110...
14360  063724  000000                  0
14361  063726  000000                  0
14362  063730  000600          DDP4:   600                             ;EXP=3
14363  063732  000000                  0                               ;FRAC=...100...
14364  063734  000000                  0
14365  063736  000000                  0
14366  063740  101100          DDP5:   101100                          ;-DDP3
14367  063742  000000                  0
14368  063744  000000                  0
```

```
14369  063746  000000                          0
14370  063750  100600          DDP6:           100600                  ;-DDP4
14371  063752  000000                          0
14372  063754  000000                          0
14373  063756  000000                          0
14374  063760  001000          DDP7:           1000                    ;DDP3+DDP6
14375  063762  000000                          0
14376  063764  000000                          0
14377  063766  000000                          0
14378  063770  101000          DDP8:           101000                  ;DDP5+DDP4
14379  063772  000000                          0
14380  063774  000000                          0
14381  063776  000000                          0
14382  064000  100400          DDP9:           100400                  ;DDP1+DDP1
14383  064002  000000                          0
14384  064004  000000                          0
14385  064006  000000                          0
14386  064010                  DDDONE:
14387  064010  004767  034164                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
14388                                          ;SEE IF THE USER HAS EXPRESSED
14389                                          ;THE DESIRE TO CHANGE THE SOFTWARE
14390                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
14391                                          ;THE USER TYPED CONTROL G?).
14392  ;*****************************************************************************
14393  ;TEST 454       SUBD TEST
14394  ;*****************************************************************************
14395  064014          TS454:
14396                  ;       USE POSITIVE OPERANDS
14397  064014  012704  003200                  MOV     #3200,R4                ;SET FIU, FIV, AND FD
14398  064020  170104                          LDFPS   R4
14399  064022  012700  064206                  MOV     #EEP1,R0                ;SET AC0 OPERAND
14400  064026  172410                          LDD     (R0),AC0
14401  064030  012700  064206                  MOV     #EEP1,R0                ;FSPC
14402  064034  173010          EE2:            SUBD    (R0),AC0                ;TEST INSTRUCTION
14403  064036  170205                          STFPS   R5                      ;GET FPS
14404  064040  012700  064164                  MOV     #EEDAT0,R0              ;GET THE RESULT
14405  064044  174010                          STD     AC0,(R0)
14406  064046  012701  064174                  MOV     #EEP0,R1                ;IS IT CORRECT?
14407  064052  012702  000004                  MOV     #4,R2
14408  064056  022021          EE3:            CMP     (R0)+,(R1)+
14409  064060  001401                          BEQ     EE6
14410  064062  104000                          EMT                     ;
14411  064064  077204          EE6:            SOB     R2,EE3
14412  064066  052704  000004                  BIS     #4,R4
14413  064072  020405                          CMP     R4,R5                   ;FPS CORRECT?
14414  064074  001401                          BEQ     EE7
14415  064076  104000                          EMT                     ;
14416                                          ;USE NEGATIVE OPERANDS
14417  064100  012704  003200          EE7:    MOV     #3200,R4                ;SET FIO, FIV, AND FD
14418  064104  170104                          LDFPS   R4
14419  064106  012700  064226                  MOV     #EEP3,R0                ;SET AC0 OPERAND
14420  064112  172410                          LDD     (R0),AC0
14421  064114  012700  064226                  MOV     #EEP3,R0                ;FSPC
14422  064120  173010          EE8:            SUBD    (R0),AC0                ;TEST INSTRUCTION
14423  064122  170205                          STFPS   R5                      ;GET FPS
14424  064124  012700  064164                  MOV     #EEDAT0,R0              ;GET THE RESULT
```

```
14425  064130  174010                  STD     AC0,(R0)
14426  064132  012701  064174          MOV     #EEP0,R1                ;IS IT CORRECT?
14427  064136  012702  000004          MOV     #4,R2
14428  064142  022021                  EE9:    CMP     (R0)+,(R1)+
14429  064144  001401                  BEQ     EE12
14430  064146  104000                  EMT                             ;
14431  064150  077204                  EE12:   SOB     R2,EE9
14432  064152  052704  000004          BIS     #4,R4
14433  064156  020405                  CMP     R4,R5                   ;FPS CORRECT?
14434  064160  001432                  BEQ     EEDONE
14435  064162  104000                  EMT                             ;
14436  064164  000000                  EEDATO: 0
14437  064166  000000                          0
14438  064170  000000                          0
14439  064172  000000                          0
14440  064174  000000                  EEP0:   0
14441  064176  000000                          0
14442  064200  000000                  00000
14443  064202  000000                          0
14444  064204  000000                          0
14445  064206  000200                  EEP1:   200
14446  064210  000000                          0
14447  064212  000000                          0
14448  064214  000000                          0
14449  064216  000400                  EEP2:   400
14450  064220  000000                          0
14451  064222  000000                          0
14452  064224  000000                          0
14453  064226  100200                  EEP3:   100200
14454  064230  000000                          0
14455  064232  000000                          0
14456  064234  000000                          0
14457  064236  100400                  EEP4:   100400
14458  064240  000000                          0
14459  064242  000000                          0
14460  064244  000000                          0
14461  064246                          EEDONE:
14462  064246  004767  033726                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
14463                                                                  ;SEE IF THE USER HAS EXPRESSED
14464                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
14465                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
14466                                                                  ;THE USER TYPED CONTROL G?).
14467                                  ;*****************************************************************************
14468                                  ;TEST 455       NORMALIZE ALGORITHM TEST
14469                                  ;*****************************************************************************
14470  064252                          TS455:
14471                                  ;USE DATA PATTERNS THAT REQUIRE ONLY ONE LEFT SHIFT TO NORMALIZE
14472  064252  012704  003200                  MOV     #3200,R4                ;SET FIO, FIV, AND FD
14473  064256  170104                          LDFPS   R4
14474  064260  012700  064442                  MOV     #FFP2,R0                ;SET AC0 OPERAND
14475  064264  172410                          LDD     (R0),AC0
14476  064266  012700  064452                  MOV     #FFP3,R0                ;FSPC
14477  064272  172010                  FF2:    ADDD    (R0),AC0                ;TEST INSTRUCTION
14478  064274  170205                          STFPS   R5                      ;GET FPS
14479  064276  012700  064412                  MOV     #FFDATO,R0              ;GET THE RESULT
14480  064302  174010                          STD     AC0,(R0)
```

```
 14481  064304  012701  064462          MOV     #FFP4, R1               ;IS IT CORRECT
```

```
14482  064310  012702  000004          MOV     #4,R2
14483  064314  022021          FF3:    CMP     (R0)+,(R1)+
14484  064316  001401                  BEQ     FF4
14485  064320  104000                  EMT                             ;
14486  064322  077204          FF4:    SOB     R2,FF3
14487  064324  020405                  CMP     R4,R5                   ;FPS CORRECT?
14488  064326  001401                  BEQ     FF5
14489  064330  104000                  EMT                             ;
14490                          ;USE DATA PATTERNS WHICH REQUIRE 56 LEFT SHIFTS TO NORMALIZE
14491                          ;THE RESULT
14492  064332  012704  003200  FF5:    MOV     #3200,R4                ;SET FIU, FIV, AND FD
14493  064336  170104                  LDFPS   R4
14494  064340  012700  064422          MOV     #FFP0,R0                ;SET AC0 OPERAND
14495  064344  172410                  LDD     (R0),AC0
14496  064346  012700  064432          MOV     #FFP1,R0                ;FSRC
14497  064352  172010          FF6:    ADDD    (R0),AC0                ;TEST INSTRUCTION
14498  064354  170205                  STFPS   R5                      ;GET FPS
14499  064356  012700  064412          MOV     #FFDAT0,R0              ;GET THE RESULT
14500  064362  174010                  STD     AC0,(R0)
14501  064364  012701  064462          MOV     #FFP4,R1                ;IS IT CORRECT.
14502  064370  012702  000004          MOV     #4,R2
14503  064374  022021          FF7:    CMP     (R0)+,(R1)+
14504  064376  001401                  BEQ     FF10
14505  064400  104000                  EMT                             ;
14506  064402  077204          FF10:   SOB     R2,FF7
14507  064404  020405                  CMP     R4,R5                   ;FPS CORRECT?
14508  064406  001431                  BEQ     FFDONE
14509  064410  104000                  EMT                             ;
14510
14511
14512  064412  000000          FFDAT0: 0
14513  064414  000000                  0
14514  064416  000000                  0
14515  064420  000000                  0
14516
14517  064422  016000          FFP0:   16000
14518  064424  000000                  0
14519  064426  000000                  0
14520  064430  000001                  1
14521  064432  116000          FFP1:   116000
14522  064434  000000                  0
14523  064436  000000                  0
14524  064440  000000                  0
14525  064442  000500          FFP2:   500
14526  064444  000000                  0
14527  064446  000000                  0
14528  064450  000000                  0
14529  064452  100400          FFP3:   100400
14530  064454  000000                  0
14531  064456  000000                  0
14532  064460  000000                  0
14533  064462  000200          FFP4:   200                             ;FFP4=FFP0+FFP1
14534  064464  000000                  0                               ;     =FFP3+FFP4
14535  064466  000000                  0
14536  064470  000000                  0
14537
```

```
14538  064472
14539  064472  004767  033502          FFDONE:
14540                                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
14541                                                                  ;SEE IF THE USER HAS EXPRESSED
14542                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
14543                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
14544                                                                  ;THE USER TYPED CONTROL G?).
14545
14546
14547                                  ;*****************************************************************************
14548                                  ;*****************************************************************************
14549                                  ;FLOATING POINT SECOND PART
14550                                  ;*****************************************************************************
14551                                  ;*****************************************************************************
14552
14553
14554                                  ;*************************************************************************************************
14555                                  ;TEST 456       ROUND\TRUNK TEST
14556                                  ;*************************************************************************************************
14557  064476                          TS456:
14558
14559                                          ;ROUND AND NORMALIZE TEST
14560
14561  064476  012704  003200                  MOV     #3200,R4                ;SET FIU, FIV, AND FD
14562  064502  170104                          LDFPS   R4
14563  064504  012700  065106                  MOV     #HHP0,R0                ;SET AC0 OPERAND
14564  064510  172410                          LDD     (R0),AC0
14565  064512  012700  065116                  MOV     #HHP1,R0                ;FSPC
14566  064516  172010                  HH2:    ADDD    (R0),AC0                ;TEST INSTRUCTION
14567  064520  170205                          STFPS   R5                      ;GET FPS
14568  064522  012700  065076                  MOV     #HHDAT0,R0              ;GET THE RESULT
14569  064526  174010                          STD     AC0,(R0)
14570  064530  012701  065126                  MOV     #HHP2,R1                ;IS IT CORRECT
14571  064534  012702  000004                  MOV     #4,R2
14572  064540  022021                  HH3:    CMP     (R0)+,(R1)+
14573  064542  001401                          BEQ     HH6
14574  064544  104000                          EMT                     ;
14575  064546  077204                  HH6:    SOB     R2,HH3
14576  064550  020405                          CMP     R4,R5           ;FPS CORRECT?
14577  064552  001401                          BEQ     HH7
14578  064554  104000                          EMT                     ;
14579
14580                                  ;THIS IS A TEST OF THE ABILITY
14581                                  ;OF NORMALIZE TO PRODUCE A ZERO EXP. AND
14582                                  ;OF THE R\T ALGORITHM TO PROPERLY SET THE FPS
14583
14584  064556  012704  043200          HH7:    MOV     #043200,R4              ;SET FIU,FIV,AND FD
14585                                                                          ;FID
14586  064562  170104                          LDFPS   R4
14587  064564  012700  065156                  MOV     #HHP5,R0                ;SET AC0 OPERAND
14588  064570  172410                          LDD     (R0),AC0
14589  064572  012700  065166                  MOV     #HHP6,R0                ;FSPC
14590  064576  172010                  HH8:    ADDD    (R0),AC0                ;TEST INSTRUCTION
14591  064600  170205                          STFPS   R5                      ;GET FPS
14592  064602  012700  065076                  MOV     #HHDAT0,R0              ;GET THE RESULT
14593  064606  174010                          STD     AC0,(R0)
```

```
14594  064610  012701  065146          MOV     #HHP4,R1                ;IS IT CORRECT
14595  064614  012702  000004          MOV     #4,R2
14596  064620  022021          HH9:    CMP     (R0)+,(R1)+
14597  064622  001401                  BEQ     HH10
14598  064624  104000                  EMT                     ;
14599  064626  077204          HH10:   SOB     R2,HH9
14600  064630  052704  100004          BIS     #100004,R4
14601  064634  020405                  CMP     R4,R5
14602  064636  001401                  BEQ     HH11
14603  064640  104000                  EMT                     ;
14604
14605                           ;THIS IS A TEST OF THE R\T ALGORITHN'S
14606                           ;ABILITY TO SET BOTH N AND Z ON A - 0 RESULT
14607  064642  012704  043200  HH11:   MOV     #043200,R4              ;SET FIV, FIV, AND FD
14608
14609  064646  170104          LDFPS   R4
14610  064650  012700  065206          MOV     #HHP8,R0                ;SET AC0 OPERAND
14611  064654  172410                  LDD     (R0),AC0
14612  064656  012700  065216          MOV     #HHP9,R0                ;FSPC
14613  064662  172010          HH12:   ADDD    (R0),AC0                ;TEST INSTRUCTION
14614  064664  170205                  STFPS   R5                      ;GET FPS
14615  064666  012700  065076          MOV     #HHDAT0,R0              ;GET THE RESULT
14616  064672  174010                  STD     AC0,(R0)
14617  064674  012701  065176          MOV     #HHP7,R1                ;IS IT CORRECT
14618  064700  012702  000004          MOV     #4,R2
14619  064704  022021          HH13:   CMP     (R0)+,(R1)+
14620  064706  001401                  BEQ     HH16
14621  064710  104000                  EMT                     ;
14622  064712  077204          HH16:   SOB     R2,HH13
14623  064714  052704  100014          BIS     #100014,R4              ;FPS CORRECT?
14624  064720  020405                  CMP     R4,R5
14625  064722  001401                  BEQ     HH17
14626  064724  104000                  EMT                     ;
14627                           ;TEST THAT CC ARE CLEARED BY R\T
14628  064726  012704  000200  HH17:   MOV     #00200,R4               ;SET FIV, FIV, AND FD
14629  064732  170104                  LDFPS   R4
14630  064734  012700  065206          MOV     #HHP8,R0                ;SET AC0 OPERAND
14631  064740  172410                  LDD     (R0),AC0
14632  064742  012700  065206          MOV     #HHP8,R0                ;FSPC
14633  064746  172010          HH18:   ADDD    (R0),AC0                ;TEST INSTRUCTION
14634  064750  170205                  STFPS   R5                      ;GET FPS
14635  064752  012700  065076          MOV     #HHDAT0,R0              ;GET THE RESULT
14636  064756  174010                  STD     AC0,(R0)
14637  064760  012701  065226          MOV     #HHP10,R1               ;IS IT CORRECT
14638  064764  012702  000004          MOV     #4,R2
14639  064770  022021          HH19:   CMP     (R0)+,(R1)+
14640  064772  001401                  BEQ     HH20
14641  064774  104000                  EMT                     ;
14642  064776  077204          HH20:   SOB     R2,HH19
14643  065000  052704  000000          BIS     #00000,R4               ;FPS CORRECT?
14644  065004  020405                  CMP     R4,R5
14645  065006  001401                  BEQ     HH21
14646  065010  104000                  EMT                     ;
14647                           ;TEST THAT N IS SET BY R\T
14648  065012  012704  003200  HH21:   MOV     #3200,R4                ;SET FIV, FIV, AND FD
14649  065016  170104                  LDFPS   R4
```

```
14650  065020  012700  065156          MOV     #HHP5,R0                ;SET AC0 OPERAND
14651  065024  172410                  LDD     (R0),AC0
14652  065026  012700  065156          MOV     #HHP5,R0                ;FSPC
14653  065032  172010          HH22:   ADDD    (R0),AC0                ;TEST INSTRUCTION
14654  065034  170205                  STFPS   R5                      ;GET FPS
14655  065036  012700  065076          MOV     #HHDATO,R0              ;GET THE RESULT
14656  065042  174010                  STD     AC0,(R0)
14657  065044  012701  065236          MOV     #HHP11,R1               ;IS IT CORRECT
14658  065050  012702  000004          MOV     #4,R2
14659  065054  022021          HH23:   CMP     (R0)+,(R1)+
14660  065056  001401                  BEQ     HH24
14661  065060  104000                  EMT                     ;
14662  065062  077204          HH24:   SOB     R2,HH23
14663  065064  052704  000010          BIS     #10,R4
14664  065070  020405                  CMP     R4,R5                   ;FPS CORRECT?
14665  065072  001465                  BEQ     HHDONE
14666  065074  104000                  EMT                     ;
14667  065076  000000          HHDATO: 0
14668  065100  000000                  0
14669  065102  000000                  0
14670  065104  000000                  0
14671  065106  000452          HHP0:   452
14672  065110  125252                  125252
14673  065112  125252                  125252
14674  065114  125253                  125253
14675  065116  000252          HHP1:   252
14676  065120  125252                  125252
14677  065122  125252                  125252
14678  065124  125252                  125252
14679  065126  000600          HHP2:   600                     ;HHP0 + HHP1 WITH
14680  065130  000000                  0                       ;PROPER NORMALIZATION
14681  065132  000000                  0
14682  065134  000000                  0
14683  065136  000400          HHP3:   400                     ;HHP0 + HHP1 WITH
14684  065140  000000                  0                       ;BAD NORMALIZATION
14685  065142  000000                  0
14686  065144  000000                  0
14687  065146  000000          HHP4:   0
14688  065150  000000                  0
14689  065152  000000                  0
14690  065154  000000                  0
14691  065156  100200          HHP5:   100200
14692  065160  000000                  0
14693  065162  000000                  0
14694  065164  000000                  0
14695  065166  000300          HHP6:   300
14696  065170  000000                  0
14697  065172  000000                  0
14698  065174  000000                  0
14699  065176  100000          HHP7:   100000                  ;HHP7 = HHP8 + HHP9
14700  065200  000000                  0                       ;      = HHP5 + HHP6
14701  065202  000000                  0
14702  065204  000000                  0
14703  065206  000200          HHP8:   200
14704  065210  000000                  0
14705  065212  000000                  0
```

```
14706  065214  000000                          0
14707  065216  100300                  HHP9:   100300
14708  065220  000000                          0
14709  065222  000000                          0
14710  065224  000000                          0
14711  065226  000400                  HHP10:  400                     ;HHP10 = HHP8 + HHP8
14712  065230  000000                          0
14713  065232  000000                          0
14714  065234  000000                          0
14715  065236  100400                  HHP11:  100400                  ;HHP11 = HHP5 + HHP5
14716  065240  000000                          0
14717  065242  000000                          0
14718  065244  000000                          0
14719  065246                          HHDONE:
14720  065246  004767  032726                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
14721                                                                  ;SEE IF THE USER HAS EXPRESSED
14722                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
14723                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
14724                                                                  ;THE USER TYPED CONTROL G?).
14725
14726
14727                                   ;*********************************************************************************
14728                                   ;TEST 457       OVER\UNDER TEST
14729                                   ;*********************************************************************************
14730  065252                          TS457:
14731
14732                                   ;TEST OVERFLOW CONDITION WITH TRAP DISABLER FIV=0
14733  065252  012704  000200                  MOV     #200,R4         ;CLEAR FIU, FIV, AND SET FD
14734  065256  170104                          LDFPS   R4
14735  065260  012737  065376  000244          MOV     #GGER0,@#FPVECT
14736  065266  012700  066200                  MOV     #GGP5,R0        ;SET AC0 OPERAND
14737  065272  172410                          LDD     (R0),AC0
14738  065274  012700  066200                  MOV     #GGP5,R0        ;FSRC
14739  065300  172010                  GG2:    ADDD    (R0),AC0        ;TEST INSTRUCTION
14740  065302  170205                          STFPS   R5              ;GET FPS
14741  065304  012700  066130                  MOV     #GGDAT0,R0      ;GET THE RESULT
14742  065310  174010                          STD     AC0,(R0)
14743  065312  012701  066210                  MOV     #GGP6,R1        ;IS IT CORRECT
14744  065316  012702  000004                  MOV     #4,R2
14745  065322  022021                  GG3:    CMP     (R0)+,(R1)+
14746  065324  001401                          BEQ     GG4
14747  065326  104000                          EMT                     ;
14748  065330  077204                  GG4:    SOB     R2,GG3
14749  065332  052704  000006                  BIS     #6,R4           ;FPS CORRECT?
14750  065336  020405                          CMP     R4,R5
14751  065340  001401                          BEQ     GG5
14752  065342  104000                          EMT                     ;
14753                                   ;TEST OVERFLOW WITH TRAPS ENABLED
14754                                   ;FIV = 1
14755  065344  012704  001200          GG5:    MOV     #1200,R4        ;CLEAR FIU, SET FIV, AND FD
14756  065350  170104                          LDFPS   R4
14757  065352  012737  065400  000244          MOV     #GG7,@#FPVECT
14758  065360  012700  066200                  MOV     #GGP5,R0        ;SET AC0 OPERAND
14759  065364  172410                          LDD     (R0),AC0
14760  065366  012700  066200                  MOV     #GGP5,R0        ;FSPC
14761  065372  172010                  GG6:    ADDD    (R0),AC0        ;TEST INSTRUCTION
```

```
14762  065374  170000                          CFCC                    ;NO OVERFLOW TRAP OCCURED
14763  065376                          GGER0:
14764  065376  104000                          EMT                     ;
14765  065400  012703  065374        GG7:      MOV     #GG6+2,R3
14766  065404  020316                          CMP     R3,(SP)         ;CHECK STACK DATA
14767  065406  001401                          BEQ     1$
14768  065410  104000                          EMT                     ;
14769  065412  022626                  1$:     CMP     (SP)+,(SP)+
14770  065414  170205                          STFPS   R5
14771  065416  012700  066130                  MOV     #GGDAT0,R0      ;GET THE RESULT .
14772  065422  174010                          STD     AC0,(R0)
14773  065424  012701  066210                  MOV     #GGP6,R1        ;IS IT CORRECT
14774  065430  012702  000004                  MOV     #4,R2
14775  065434  022021                  GG8:    CMP     (R0)+,(R1)+
14776  065436  001401                          BEQ     GG9
14777  065440  104000                          EMT                     ;
14778  065442  077204                  GG9:    SOB     R2,GG8
14779  065444  052704  100006                  BIS     #100006,R4      ;EXACT ZERO RESULTED IF OVERFLOW
14780  065450  020405                          CMP     R4,R5           ;FPS CORRECT?, CHECK FER, FZ, FV
14781  065452  001401                          BEQ     1$
14782  065454  104000                          EMT                     ;
14783  065456  012704  000010          1$:     MOV     #10,R4
14784                                  ;CHECK FEC
14785  065462  170305                          STST    R5
14786  065464  020405                          CMP     R4,R5
14787  065466  001401                          BEQ     GG10
14788  065470  104000                          EMT                     ;
14789                                  ;CHECK UNDER FLOW CONDITION WITH
14790                                  ;TRAPS DISABLED (FIU = 0)
14791  065472  012704  000200          GG10:   MOV     #0200,R4        ;SET FIU, FIV, AND FD
14792  065476  170104                          LDFPS   R4
14793  065500  012737  065376  000244        MOV     #GGER0,@#FPVECT
14794  065506  012700  066150                MOV     #GGP2,R0        ;SET AC0 OPERAND
14795  065512  172410                        LDD     (R0),AC0        ;FSRC
14796  065514  012700  066160                MOV     #GGP3,R0
14797  065520  172010                  GG11:   ADDD    (R0),AC0        ;TEST INSTRUCTION
14798  065522  170205                        STFPS   R5              ;GET FPS
14799  065524  012700  066130                MOV     #GGDAT0,R0      ;GET THE RESULT
14800  065530  174010                        STD     AC0,(R0)
14801  065532  012701  066210                MOV     #GGP6,R1        ;IS IT CORRECT
14802  065536  012702  000004                MOV     #4,R2
14803  065542  022021                  GG12:   CMP     (R0)+,(R1)+
14804  065544  001401                        BEQ     GG13
14805  065546  104000                        EMT                     ;
14806  065550  077204                  GG13:   SOB     R2,GG12
14807  065552  052704  000004                BIS     #4,R4           ;FPS CORRECT?
14808  065556  020405                        CMP     R4,R5
14809  065560  001401                        BEQ     GG14
14810  065562  104000                        EMT                     ;
14811                                  ;CHECK UNDERFLOW CONDITION WITH
14812                                  ;TRAP ENABLED (FIU = 1)
14813  065564  012704  002200          GG14:   MOV     #2200,R4        ;SET FIU, FIV, AND FD
14814  065570  170104                        LDFPS   R4
14815  065572  012737  065616  000244        MOV     #GG16,@#FPVECT
14816  065600  012700  066150                MOV     #GGP2,R0        ;SET AC0 OPERAND
14817  065604  172410                        LDD     (R0),AC0        ;FSPC
```

```
14818  065606  012700  066160          MOV     #GGP3,R0
14819  065612  172010          GG15:   ADDD    (R0),AC0        ;TEST INSTRUCTION
14820  065614  170000                  CFCC
14821  065616  012703  065614  GG16:   MOV     #GG15+2,R3
14822  065622  021603                  CMP     (SP),R3
14823  065624  001401                  BEQ     1$
14824  065626  104000                  EMT                     ;
14825  065630  022626          1$:     CMP     (SP)+,(SP)+
14826  065632  170205                  STFPS   R5              ;GET FPS
14827  065634  012700  066130          MOV     #GGDAT0,R0      ;GET THE RESULT
14828  065640  174010                  STD     AC0,(R0)
14829  065642  012701  066220          MOV     #GGP7,R1        ;IS IT CORRECT
14830  065646  012702  000004          MOV     #4,R2
14831  065652  022021          GG17:   CMP     (R0)+,(R1)+
14832  065654  001401                  BEQ     GG18
14833  065656  104000                  EMT                     ;
14834  065660  077204          GG18:   SOB     R2,GG17
14835  065662  052704  100000          BIS     #100000,R4
14836  065666  020405                  CMP     R4,R5           ;FPS CORRECT?
14837  065670  001401                  BEQ     1$
14838  065672  104000                  EMT                     ;
14839  065674  012704  000012  1$:     MOV     #12,R4
14840                          ;CHECK FEC
14841  065700  170305                  STST    R5
14842  065702  020405                  CMP     R4,R5
14843  065704  001401                  BEQ     GG19
14844  065706  104000                  EMT                     ;
14845                          ;CHECK UNDERFLOW CONDITION WITH TRAPS
14846                          ;DISABLED (FIU = 0)
14847  065710  012704  000200  GG19:   MOV     #0200,R4        ;SET FIU, FIV, AND FD
14848  065714  170104                  LDFPS   R4
14849  065716  012737  066034  000244        MOV     #GGER14,@#FPVECT
14850  065724  012700  066150          MOV     #GGP2,R0        ;SET AC0 OPERAND
14851  065730  172410                  LDD     (R0),AC0
14852  065732  012700  066230          MOV     #GGP8,R0        ;FSPC
14853  065736  172010          GG20:   ADDD    (R0),AC0        ;TEST INSTRUCTION
14854  065740  170205                  STFPS   R5              ;GET FPS
14855  065742  012700  066130          MOV     #GGDAT0,R0      ;GET THE RESULT
14856  065746  174010                  STD     AC0,(R0)
14857  065750  012701  066210          MOV     #GGP6,R1        ;IS IT CORRECT
14858  065754  012702  000004          MOV     #4,R2
14859  065760  022021          GG21:   CMP     (R0)+,(R1)+
14860  065762  001401                  BEQ     GG22
14861  065764  104000                  EMT                     ;
14862  065766  077204          GG22:   SOB     R2,GG21
14863  065770  052704  000004          BIS     #4,R4           ;FPS CORRECT?
14864  065774  020405                  CMP     R4,R5
14865  065776  001401                  BEQ     GG23
14866  066000  104000                  EMT                     ;
14867                          ;CHECK UNDERFLOW CONDITION WITH TRAP
14868                          ;ENABLED (FIU = 1)
14869  066002  012704  002200  GG23:   MOV     #2200,R4        ;SET FIU, FIV, AND FD
14870  066006  170104                  LDFPS   R4
14871  066010  012737  066036  000244        MOV     #GG25,@#FPVECT
14872  066016  012700  066150          MOV     #GGP2,R0        ;SET AC0 OPERAND
14873  066022  172410                  LDD     (R0),AC0
```

```
14874  066024  012700  066230          MOV     #GGP8,R0        ;FSRC
14875  066030  172010          GG24:   ADDD    (R0),AC0        ;TEST INSTRUCTION
14876  066032  170000                  CFCC
14877  066034                  GGER14:
14878  066034  104000                  EMT                     ;
14879  066036  012703  066032  GG25:   MOV     #GG24+2,R3
14880  066042  020316                  CMP     R3,(SP)
14881  066044  001401                  BEQ     1$
14882  066046  104000                  EMT                     ;
14883  066050  022626          1$:     CMP     (SP)+,(SP)+
14884  066052  170205                  STFPS   R5              ;GET FPS
14885  066054  012700  066130          MOV     #GGDATO,R0      ;GET THE RESULT
14886  066060  174010                  STD     AC0,(R0)
14887  066062  012701  066240          MOV     #GGP9,R1        ;IS IT CORRECT
14888  066066  012702  000004          MOV     #4,R2
14889  066072  022021          GG26:   CMP     (R0)+,(R1)+
14890  066074  001401                  BEQ     GG27
14891  066076  104000                  EMT                     ;
14892  066100  077204          GG27:   SOB     R2,GG26
14893  066102  052704  100004          BIS     #100004,R4
14894  066106  020405                  CMP     R4,R5           ;FPS CORRECT?
14895  066110  001401                  BEQ     1$
14896  066112  104000                  EMT                     ;
14897  066114  012704  000012  1$:     MOV     #12,R4
14898                                  ;CHECK FEC
14899  066120  170305                  STST    R5
14900  066122  020405                  CMP     R4,R5
14901  066124  001451                  BEQ     GGDONE
14902  066126  104000                  EMT                     ;
14903  066130  000000          GGDATO: 0
14904  066132  000000                  0
14905  066134  000000                  0
14906  066136  000000                  0
14907
14908  066140  000300          GGP1:   300
14909  066142  000000                  0
14910  066144  000000                  0
14911  066146  000000                  0
14912  066150  100200          GGP2:   100200
14913  066152  000000                  0
14914  066154  000000                  0
14915  066156  000000                  0
14916  066160  000200          GGP3:   200
14917  066162  000000                  0
14918  066164  000000                  0
14919  066166  000001                  1
14920  066170  010200          GGP4:   10200
14921  066172  000000                  0
14922  066174  000000                  0
14923  066176  000000                  0
14924  066200  077600          GGP5:   77600                   ;OVER FLOW = GGP5 + GGP5
14925  066202  000000                  0
14926  066204  000000                  0
14927  066206  000000                  0
14928  066210  000000          GGP6:   0                       ;OVERFLOW RESULT
14929  066212  000000                  0                       ;UNDERFLOW RESULT
```

```
14930  066214  000000                          0                       ;GGP6 = GGP4 + GGP5
14931  066216  000000                          0                       ;     = GGP3 + GGP2 (FIU = 0)
14932                                                                  ;     = GGP3 + GGP1
14933  066220  062400                  GGP7:   62400                   ;GGP7 = GGP3 + GGP2 (FIU = 1)
14934  066222  000000                          0
14935  066224  000000                          0
14936  066226  000000                          0
14937  066230  000340                  GGP8:   340
14938  066232  000000                          0
14939  066234  000000                          0
14940  066236  000000                          0
14941  066240  000100                  GGP9:   100
14942  066242  000000                          0
14943  066244  000000                          0
14944  066246  000000                          0
14945  066250                          GGDONE:
14946  066250  004767  031724                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
14947                                                                  ;SEE IF THE USER HAS EXPRESSED
14948                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
14949                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
14950                                                                  ;THE USER TYPED CONTROL G?).
14951
14952
14953                                   ;*****************************************************************************
14954                                   ;TEST 460       LDCFD AND LDCDF TEST
14955                                   ;*****************************************************************************
14956  066254                          TS460:
14957                                  ;TEST FOR CORRECT AUTO INCREMENT CONSTANT.
14958  066254  012704  000200                  MOV     #200,R4         ;SET LONG INTEGER MODE
14959  066260  170104                          LDFPS   R4
14960  066262  012700  067012                  MOV     #HXP1,R0
14961  066266  172410                          LDD     (R0),AC0
14962  066270  012700  067022                  MOV     #HXP2,R0
14963  066274  177420                  HX2:    LDCFD   (R0)+,AC0
14964  066276  020027  067026                  CMP     R0,#HXP2+4      ;IS R0 CORRECT
14965  066302  001401                          BEQ     HX3
14966  066304  104000                          EMT                     ;
14967  066306                          HX3:
14968  066306  170205                          STFPS   R5              ;GET FPS
14969  066310  012700  067002                  MOV     #HXDAT0,R0
14970  066314  174010                          STD     AC0,(R0)        ;GET AC0
14971  066316  012701  067072                  MOV     #HXP7,R1        ;SEE IF RESULT IS
14972  066322  012702  000004                  MOV     #4,R2           ;CORRECT
14973  066326  022120                  HX4:    CMP     (R1)+,(R0)+
14974  066330  001401                          BEQ     HX7
14975  066332  104000                          EMT                     ;
14976  066334  077204                  HX7:    SOB     R2,HX4
14977  066336  012704  000200                  MOV     #200,R4         ;FPS CORRECT?
14978  066342  020405                          CMP     R4,R5
14979  066344  001401                          BEQ     HX8
14980  066346  104000                          EMT                     ;
14981                                  ;NOW    TEST    LDCDF
14982  066350  012704  000200          HX8:    MOV #200,R4
14983  066354  170104                          LDFPS R4
14984
14985  066356  012700  067012                  MOV     #HXP1,R0
```

```
14986  066362  172410                          LDD     (R0),AC0
14987
14988  066364  012700  067022                  MOV     #HXP2,R0
14989  066370  170001                          SETF
14990  066372  177420                  HX9:    LDCDF   (R0)+,AC0       ;TEST INSTRUCTION
14991  066374  020027  067032                  CMP     R0,#HXP2+10     ;WAS A GOOD
14992  066400  001401                          BEQ     HX10
14993  066402  104000                          EMT                     ;
14994
14995  066404                          HX10:
14996  066404  170205                          STFPS   R5
14997  066406  012700  067002                  MOV     #HXDATO,R0
14998  066412  170011                          SETD
14999  066414  174010                          STD     AC0,(R0)        ;GET RESULT
15000  066416  012701  067102                  MOV     #HXP8,R1
15001  066422  012702  000004                  MOV     #4,R2
15002  066426  022120                  HX11:   CMP     (R1)+,(R0)+     ;IS IT CORRECT?
15003  066430  001401                          BEQ     HX14
15004  066432  104000                          EMT                     ;
15005  066434  077204                  HX14:   SOB     R2,HX11
15006
15007  066436  012704  000000                  MOV     #0,R4           ;FPS CORRECT?
15008  066442  020405                          CMP     R4,R5
15009  066444  001401                          BEQ     HX15
15010  066446  104000                          EMT                     ;
15011                                  ;TEST GR7 IMMEDIATE MODE CONSTANT
15012  066450  012704  000200          HX15:   MOV     #200,R4
15013  066454  170104                          LDFPS   R4              ;SET FD
15014  066456  012737  066506  000004          MOV     #HXER9,@#ERRVECT
15015  066464  005001                          CLR     R1
15016  066466  177427  043243          HX16:   LDCFD   #5201,AC0
15017  066472  005201                  HX165:  INC     R1
15018  066474  005201                          INC     R1
15019  066476  005201                          INC     R1
15020  066500  020127  000003                  CMP     R1,#3           ;SEE IF PC WAS
15021  066504  001401                          BEQ     HX17
15022  066506                          HXER9:
15023  066506  104000                          EMT                     ;
15024  066510  012704  000200          HX17:   MOV     #200,R4
15025  066514  170104                          LDFPS   R4
15026  066516  012700  067062                  MOV     #HXP6,R0
15027  066522  172410                          LDD     (R0),AC0
15028  066524  012700  067022                  MOV     #HXP2,R0
15029  066530  177410                  HX18:   LDCFD   (R0),AC0
15030
15031  066532  012700  067002                  MOV     #HXDATO,R0
15032  066536  174010                          STD     AC0,(R0)        ;GET RESULT.
15033  066540  012701  067072                  MOV     #HXP7,R1
15034  066544  012702  000004                  MOV     #4,R2
15035  066550  022021                  HX19:   CMP     (R0)+,(R1)+     ;IS RESULT CORRECT?
15036  066552  001401                          BEQ     HX20
15037  066554  104000                          EMT                     ;
15038  066556  077204                  HX20:   SOB     R2,HX19
15039
15040                                  ;TEST LDCFD WITH NEGATIVE OPERAND
15041  066560  012704  000200                  MOV     #200,R4
```

```
15042  066564  170104                    LDFPS   R4
15043  066566  012700  067062            MOV     #HXP6,R0
15044  066572  172410                    LDD     (R0),AC0
15045  066574  012700  067042            MOV     #HXP4,R0
15046  066600  177410            HX22:   LDCFD   (R0),AC0
15047
15048  066602  012700  067002            MOV     #HXDAT0,R0
15049  066606  174010                    STD     AC0,(R0)        ;GET RESULT
15050
15051  066610  012701  067052            MOV     #HXP5,R1
15052  066614  012702  000004            MOV     #4,R2
15053  066620  022120            HX23:   CMP     (R1)+,(R0)+
15054  066622  001401                    BEQ     HX26
15055  066624  104000                    EMT                     ;
15056  066626  077204            HX26:   SOB     R2,HX23
15057
15058                             ;TEST   LDCFD   0
15059
15060  066630  012704  000200            MOV     #200,R4
15061  066634  170104                    LDFPS   R4
15062
15063  066636  012700  067012            MOV     #HXP1,R0
15064  066642  172410                    LDD     (R0),AC0
15065  066644  172010                    ADDD    (R0),AC0
15066
15067  066646  012700  067012            MOV     #HXP1,R0
15068  066652  177410            HX28:   LDCFD   (R0),AC0
15069
15070  066654  170205                    STFPS   R5
15071
15072  066656  012700  067002            MOV     #HXDAT0,R0
15073  066662  174010                    STD     AC0,(R0)        ;GET RESULT
15074
15075  066664  012701  067012            MOV     #HXP1,R1
15076  066670  012702  000004            MOV     #4,R2
15077  066674  022120            HX29:   CMP     (R1)+,(R0)+     ;IS IT 0?
15078  066676  001401                    BEQ     HX30
15079  066700  104000                    EMT                     ;
15080  066702  077204            HX30:   SOB     R2,HX29
15081
15082  066704  012704  000204            MOV     #204,R4         ;FPS CORRECT
15083  066710  020405                    CMP     R4,R5
15084  066712  001401                    BEQ     HX31
15085  066714  104000                    EMT                     ;
15086                             ;TEST   LDCFD   0
15087  066716  012704  000200    HX31:   MOV     #200,R4
15088  066722  170104                    LDFPS   R4
15089  066724  012700  067062            MOV     #HXP6,R0
15090  066730  172410                    LDD     (R0),AC0
15091  066732  012700  067012            MOV     #HXP1,R0
15092  066736  177410            HX32:   LDCFD   (R0),AC0
15093  066740  170205                    STFPS   R5
15094  066742  012700  067002            MOV     #HXDAT0,R0
15095  066746  174010                    STD     AC0,(R0)        ;GET RESULT
15096  066750  012701  067012            MOV     #HXP1,R1
15097  066754  012702  000004            MOV     #4,R2
```

```
15098  066760  022120                HX33:   CMP     (R1)+,(R0)+     ;IS IT ZERO?
15099  066762  001401                        BEQ     HX34
15100  066764  104000                        EMT                     ;
15101  066766  077204                HX34:   SOB     R2,HX33
15102
15103  066770  012704  000204              MOV     #204,R4         ;FPS CORRECT?
15104  066774  020405                        CMP     R4,R5
15105  066776  001445                        BEQ     HXDONE
15106  067000  104000                        EMT                     ;
15107
15108  067002  000000                HXDATO: 0
15109  067004  000000                        0
15110  067006  000000                        0
15111  067010  000000                        0
15112
15113  067012  000000                HXP1:   0
15114  067014  000000                        0
15115  067016  000000                        0
15116  067020  000000                        0
15117
15118  067022  000577                HXP2:   577
15119  067024  177776                        177776
15120  067026  177777                        177777
15121  067030  177776                        177776
15122  067032  005201                HXP3:   5201
15123  067034  000000                        0
15124  067036  000000                        0
15125  067040  000000                        0
15126  067042  100577                HXP4:   100577
15127  067044  177776                        177776
15128  067046  177777                        177777
15129  067050  177776                        177776
15130  067052  100577                HXP5:   100577
15131  067054  177776                        177776
15132  067056  000000                        0
15133  067060  000000                        0
15134  067062  000252                HXP6:   252
15135  067064  125252                        125252
15136  067066  125252                        125252
15137  067070  125252                        125252
15138
15139  067072  000577                HXP7:   577
15140  067074  177776                        177776
15141  067076  000000                        0
15142  067100  000000                        0
15143  067102  000577                HXP8:   577
15144  067104  177777                        177777
15145  067106  000000                        0
15146  067110  000000                        0
15147
15148  067112                        HXDONE:
15149  067112  004767  031062                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
15150                                                                ;SEE IF THE USER HAS EXPRESSED
15151                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
15152                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
15153                                                                ;THE USER TYPED CONTROL G?).
```

```
15154
15155
15156
15157
15158                                   ;*****************************************************************************************
15159                                   ;TEST 461       CMPD TEST
15160                                   ;*****************************************************************************************
15161  067116                           TS461:
15162
15163                                   ;TEST THE CMPD INSTRUCTION WITH (FSRC=AC=0)
15164  067116  004737  067612           AAA1:   JSR     PC,@#CMPSUB
15165  067122  000000  000000  000000   1$:     .WORD   0,0,0,0                 ;AC0
15166  067130  000000
15167  067132  000000  000000  000000   2$:     .WORD   0,0,0,0                 ;FSRC
15168  067140  000000
15169  067142  000200                   3$:     200                             ;FPS BEFORE EXECUTION
15170  067144  000204                           204                             ;FPS AFTER EXECUTION
15171
15172
15173                                   ;TEST CMPD WITH (AC=0) AND FSRC POSITIVE.
15174  067146  00'737  067612           AAA2:   JSR     PC,@#CMPSUB
15175  067152  0C0000  000000  000000   1$:     .WORD   0,0,0,0                 ;AC
15176  067160  000000
15177  067162  025252                   2$:     25252                           ;FSRC
15178  067164  052525                           52525
15179  067166  125252                           125252
15180  067170  052525                           52525
15181  067172  000200                   3$:     200                             ;FPS BEFORE EXECUTION
15182  067174  000200                           200                             ;FPS AFTER EXECUTION
15183
15184                                   ;TEST CMPD WITH (AC=0) AND FSRC NEGATIVE
15185  067176  004737  067612           AAA3:   JSR     PC,@#CMPSUB
15186  067202  000000  000000  000000   1$:     .WORD   0,0,0,0                 ;AC
15187  067210  000000
15188  067212  125252                   2$:     125252                          ;FSRC
15189  067214  125252                           125252
15190  067216  052525                           52525
15191  067220  125252                           125252
15192  067222  000200                   3$:     200                             ;FPS BEFORE EXECUTION
15193  067224  000210                           210                             ;FPS AFTER EXECUTION
15194
15195                                   ;TEST CMPD WITH (FSRC=0) AND AC POSITIVE
15196  067226  004737  067612           AAA4:   JSR     PC,@#CMPSUB
15197  067232  025252                   1$:     25252                           ;AC
15198  067234  052525                           52525
15199  067236  125252                           125252
15200  067240  052525                           52525
15201  067242  000000  000000  000000   2$:     .WORD   0,0,0,0                 ;FSRC
15202  067250  000000
15203  067252  000200                   3$:     200                             ;FPS BEFORE EXECUTION
15204  067254  000210                           210                             ;FPS AFTER EXECUTION
15205
15206
15207                                   ;TEST CMPD WITH (FSRC=0) AND AC NEGATIVE
15208  067256  004737  067612           AAA5:   JSR     PC,@#CMPSUB
15209  067262  125252                   1$:     125252                          ;AC
```

```
15210  067264  125252                          125252
15211  067266  052525                          52525
15212  067270  125252                          125252
15213  067272  000000  000000  000000  2$:     .WORD   0,0,0,0                 ;FSRC
15214  067300  000000
15215  067302  000200                  3$:     200                             ;FPS BEFORE EXECUTION
15216  067304  000200                          200                             ;FPS AFTER EXECUTION
15217
15218                                  ;TEST CMPD WITH AC POSITIVE AND FSRC NEGATIVE
15219  067306  004737  067612          AAA6:   JSR     PC,@#CMPSUB
15220  067312  052525                  1$:     52525                           ;AC
15221  067314  125252                          125252
15222  067316  052525                          52525
15223  067320  125252                          125252
15224  067322  125252                  2$:     125252                          ;;FSRC
15225  067324  052525                          52525
15226  067326  125252                          125252
15227  067330  052525                          52525
15228  067332  000200                  3$:     200                             ;FPS BEFORE EXECUTION
15229  067334  000210                          210                             ;FPS AFTER EXECUTION
15230
15231
15232                                  ;TEST CMPD WITH AC NEGATIVE AND FSRC POSITIVE
15233  067336  004737  067612          AAA7:   JSR     PC,@#CMPSUB
15234  067342  125252                  1$:     125252                          ;AC
15235  067344  052525                          52525
15236  067346  125252                          125252
15237  067350  052525                          52525
15238  067352  052525                  2$:     52525                           ;FSRC
15239  067354  125252                          125252
15240  067356  052525                          52525
15241  067360  125252                          125252
15242  067362  000200                  3$:     200                             ;FPS BEFORE EXECUTION
15243  067364  000200                          200                             ;FPS AFTER EXECUTION
15244
15245                                  ;TEST CMPD WITH AC POSITIVE AND FSRC POSITIVE
15246                                  ;AND EAC LESS THAN EFSRC.
15247  067366  004737  067612          AAA8:   JSR     PC,@#CMPSUB
15248  067372  012345                  1$:     12345                           ;AC
15249  067374  067654                          67654
15250  067376  032101                          32101
15251  067400  023456                          23456
15252  067402  023456                  2$:     23456                           ;FSRC
15253  067404  076543                          76543
15254  067406  021012                          21012
15255  067410  034567                          34567
15256  067412  000200                  3$:     200                             ;FPS BEFORE EXECUTION
15257  067414  000200                          200                             ;FPS AFTER EXECUTION
15258
15259
15260                                  ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND EAC GREATER THAN EFSRC
15261  067416  004737  067612          AAA9:   JSR     PC,@#CMPSUB
15262  067422  045676                  1$:     45676                           ;AC
15263  067424  054321                          54321
15264  067426  012345                          12345
15265  067430  067654                          67654
```

```
15266  067432  034567                2$:     34567                           ;FSRC
15267  067434  065432                        65432
15268  067436  101234                        101234
15269  067440  056765                        56765
15270  067442  000200                3$:     200                             ;FPS BEFORE EXECUTION
15271  067444  000210                        210                             ;FPS AFTER EXECUTION
15272
15273                                ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE AND AC EQUAL TO FSRC
15274  067446  004737  067612        AAA10:  JSR     PC,@#CMPSUB
15275  067452  012345                1$:     12345                           ;AC
15276  067454  067012                        67012
15277  067456  034567                        34567
15278  067460  012345                        012345
15279  067462  012345                2$:     12345                           ;FSRC
15280  067464  067012                        67012
15281  067466  034567                        34567
15282  067470  012345                        012345
15283  067472  000200                3$:     200                             ;FPS BEFORE EXECUTION
15284  067474  000204                        204                             ;FPS AFTER EXECUTION
15285
15286                                ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
15287                                ;AND FSRC GREATER THAN AC.
15288  067476  004737  067612        AAA11:  JSR     PC,@#CMPSUB
15289  067502  012345                1$:     12345                           ;AC
15290  067504  067012                        67012
15291  067506  034567                        34567
15292  067510  012345                        012345
15293  067512  012345                2$:     12345                           ;FSRC
15294  067514  070123                        70123
15295  067516  045670                        45670
15296  067520  123456                        123456
15297  067522  000200                3$:     200                             ;FPS BEFORE EXECUTION
15298  067524  000200                        200                             ;FPS AFTER EXECUTION
15299
15300                                ;TEST CMPD WITH AC POSITIVE, FSRC POSITIVE, EAC EQUAL TO EFSRC,
15301                                ;AND AC GREATER THAN FSRC.
15302  067526  004737  067612        AAA12:  JSR     PC,@#CMPSUB
15303  067532  054321                1$:     54321                           ;AC
15304  067534  076543                        76543
15305  067536  021076                        21076
15306  067540  054321                        54321
15307  067542  054321                2$:     54321                           ;FSRC
15308  067544  065432                        65432
15309  067546  107654                        107654
15310  067550  032107                        32107
15311  067552  000200                3$:     200                             ;FPS BEFORE EXECUTION
15312  067554  000210                        210                             ;FPS AFTER EXECUTION
15313
15314                                ;TEST CMPD WITH AC NEGATIVE, FSRC NEGATIVE, EAC EQUAL TO EFSRC,
15315                                ;AND AC GREATER THAN FSRC
15316  067556  004737  067612        AAA13:  JSR     PC,@#CMPSUB
15317  067562  112345                1$:     112345                          ;AC
15318  067564  043210                        43210
15319  067566  076543                        76543
15320  067570  021076                        21076
15321  067572  112345                2$:     112345                          ;FSRC
```

```
15322  067574  054321                          54321
15323  067576  007654                          07654
15324  067600  032107                          32107
15325  067602  000200                  3$:     200                     ;FPS BEFORE EXECUTION
15326  067604  000210                          210                     ;FPS AFTER EXECUTION
15327
15328
15329  067606  000137  067716                  JMP     @#AAADONE       ;FINISHED CMPD TEST.
15330
15331
15332                                  ;THIS SUBROUTINE, CMPSUB, IS CALLED TO SET UP, EXECUTE
15333                                  ;AND CHECK THE RESULTS OF A CMPD INSTRUCTION.
15334                                  ;IT IS CALLED THUS:
15335                                  ;
15336                                  ;               JSR     PC,@#CMPSUB
15337                                  ;       ACARG:  .WORD   X,X,X,X         ;AC OPERAND
15338                                  ;       FSRCARG:.WORD   X,X,X,X         ;FSRC OPERAND
15339                                  ;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
15340                                  ;       FPSA:   .WORD   X               ;FPS AFTER EXECUTION
15341                                  ;       FPSE:   .WORD   X               ;ERROR FPS
15342                                  ;       ERR:    ERROR   X               ;FPS ERROR
15343                                  ;       CONT:                           ;RETURN ADDRESS
15344                                  ;
15345                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
15346                                  ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, CMPD, IS EXECUTED.
15347                                  ;AFTER THE EXECUTION THE FPS IS CHECKED AGAINST FPSA. IF IT IS A MATCH
15348                                  ;THEN THERE WAS NO ERROR AND CONTROL IS RETURNED TO CONT. IF
15349                                  ;THE FPS IS INCORRECT IT IS COMPARED WITH FPSE IN AN ATTEMPT TO ANALYSE
15350                                  ;THE FAILURE. IF THE FPS IS THE SAME AS FPSE THEN CONTROL IS
15351                                  ;RETURNED TO THE ERROR CALL AT LOCATION ERR. IF THE FPS WAS
15352                                  ;NOT CORRECT BUT DIDN'T MATCH FPSE A GENERAL ERROR IS REPORTED
15353                                  ;AND CONTROL IS PASSED TO CONT.
15354
15355  067612  012601                  CMPSUB: MOV     (SP)+,R1        ;PICK UP A POINTER TO THE
15356                                                                  ;ARGUMENTS.
15357  067614  016100  000020                  MOV     20(R1),R0       ;GET THE FPS BEFORE EXECUTION.
15358  067620  170100                          LDFPS   R0              ;LOAD IT INTO THE FPS.
15359
15360  067622  010100                          MOV     R1,R0           ;GET ADDRESS OF AC OPERAND.
15361  067624  172410                          LDD     (R0),AC0        ;LOAD AC0 OPERAND
15362
15363  067626  010100                          MOV     R1,R0           ;COMPUTE FSRC OPERAND
15364  067630  062700  000010                  ADD     #10,R0          ;ADDRESS
15365
15366  067634  000240                          NOP                     ;FOR SCOPING.
15367  067636  173410                  1$:     CMPD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
15368
15369  067640  170205                          STFPS   R5              ;SAVE FPS AFTER INSTRUCTION.
15370
15371  067642  016104  000022                  MOV     22(R1),R4       ;GET EXPECTED FPS.
15372  067646  020405                          CMP     R4,R5           ;WAS FPS CORRECT?
15373  067650  020405                          CMP     R4,R5           ;WAS FPS CORRECT?
15374  067652  001401                          BEQ     3$
15375  067654  104000                          EMT                     ;
15376  067656  012700  067706          3$:     MOV     #CMPTMP,R0      ;IF FPS WAS CORRECT MAKE SURE
15377  067662  174010                          STD     AC0,(R0)        ;AC0 WAS NOT AFFECTED BY CMPD.
```

```
15378  067664  010102                          MOV     R1,R2
15379  067666  012703  000004                  MOV     #4,R3
15380  067672  022220                  4$:     CMP     (R2)+,(R0)+
15381  067674  001401                          BEQ     5$
15382  067676  104000                          EMT                             ;
15383  067700  077304                  5$:     SOB     R3,4$
15384
15385  067702  000161  000024                  JMP     24(R1)                  ;RETURN
15386
15387  067706  000000  000000  000000  CMPTMP: .WORD   0,0,0,0
15388  067714  000000
15389
15390
15391
15392  067716                          AAADONE:
15393  067716  004767  030256                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
15394                                                                  ;SEE IF THE USER HAS EXPRESSED
15395                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
15396                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
15397                                                                  ;THE USER TYPED CONTROL G?).
15398
15399
15400
15401
15402                                  ;*****************************************************************************
15403                                  ;TEST 462       DIVD WITH (FSRC=0) AND (BUT FD) TEST
15404                                  ;*****************************************************************************
15405  067722                          TS462:
15406
15407  067722  012704  040200          BBB0:   MOV     #40200,R4       ;SET UP FPS
15408                                                                  ;WITH INTERRUPTS
15409                                                                  ;DISABLED.
15410  067726  170104                          LDFPS   R4
15411  067730  012737  067766  000244          MOV     #BBBER1,@#FPVECT;SET UP FOR ANY FP INTERRUPTS.
15412  067736  012700  070152                  MOV     #BBBP1,R0       ;SET UP AC0 = 0
15413  067742  172410                          LDD     (R0),AC0
15414  067744  012701  070152                  MOV     #BBBP1,R1       ;FSRC = 0
15415
15416  067750  174411                  BBB1:   DIVD    (R1),AC0        ;TEST INSTRUCTION
15417
15418  067752  170205                          STFPS   R5              ;GET FPS
15419  067754  170303                          STST    R3              ;GET FEC
15420
15421  067756  012704  140204                  MOV     #140204,R4      ;EXPECTED FPS.
15422  067762  020405                          CMP     R4,R5           ;IS FPS CORRECT.
15423  067764  001401                          BEQ     BBB7
15424  067766                          BBBER1:
15425  067766  104000                          EMT                     ;
15426  067770  012702  000004          BBB7:   MOV     #4,R2           ;EXPECTED FEC.
15427  067774  020203                          CMP     R2,R3           ;IS FEC CORRECT?
15428  067776  001401                          BEQ     BBB2
15429  070000  104000                          EMT                     ;
15430
15431                                  ;TEST DIVD WITH (FSRC=0) AND TRAPS DISABLED.
15432  070002  012704  040200          BBB2:   MOV     #40200,R4       ;LOAD FPS WITH TRAPS DISABLED.
15433  070006  170104                          LDFPS   R4
```

```
15434
15435  070010  012700  070162                  MOV     #BBBP2,R0       ;SET UP AC0 OPERAND (NON ZERO).
15436  070014  172410                          LDD     (R0),AC0
15437  070016  012700  070152                  MOV     #BBBP1,R0       ;FSRC=0
15438  070022  174410                  BBB3:   DIVD    (R0),AC0
15439
15440  070024  170205                          STFPS   R5              ;GET FPS.
15441  070026  170303                          STST    R3              ;GET FEC.
15442
15443  070030  012704  140200                  MOV     #140200,R4              ;EXPECTED FPS.
15444  070034  020405                          CMP     R4,R5           ;IS FPS CORRECT?
15445  070036  001401                          BEQ     1$
15446  070040  104000                          EMT                     ;
15447  070042  012702  000004          1$:     MOV     #4,R2           ;EXPECTED FEC.
15448  070046  020203                          CMP     R2,R3           ;WAS FEC CORRECT?
15449  070050  001401                          BEQ     BBB4
15450  070052  104000                          EMT                     ;
15451
15452                                  ;TEST DIVD WITH FSRC=0) AND TRAPS ENABLED.
15453  070054  012704  000200          BBB4:   MOV     #200,R4         ;SET UP FPS. TRAP ENABLED.
15454  070060  170104                          LDFPS   R4
15455
15456  070062  012700  070162                  MOV     #BBBP2,R0       ;SET UP AC0 OPERAND (NON ZERO).
15457  070066  172410                          LDD     (R0),AC0
15458
15459  070070  012737  070110  000244          MOV     #BBB6,@#FPVECT  ;SET UP FOR THE EXPECTED INTERRUPT.
15460  070076  012700  070152                  MOV     #BBBP1,R0       ;FSRC=0
15461
15462  070102  174410                  BBB5:   DIVD    (R0),AC0        ;TEST INSTRUCTION (SHOULD RESULT IN TRAP).
15463  070104  170000                          CFCC
15464  070106  104000                          EMT                     ;
15465  070110  022716  070104          BBB6:   CMP     #BBB5+2,(SP)    ;TRAP TO HERE WHEN THE DIVISION BY 0
15466                                                                  ;OCCURS. FIRST SEE IF THE ADDRESS OF
15467                                                                  ;THE TRAP IS 2+THE ADDRESS OF THE TEST
15468                                                                  ;DIVD INSTRUCTION.
15469  070114  001401                          BEQ     1$
15470  070116  104000                          EMT                     ;
15471  070120  170205                  1$:     STFPS   R5              ;GET FPS.
15472  070122  170303                          STST    R3              ;GET FEC.
15473  070124  022626                          CMP     (SP)+,(SP)+     ;RESET THE STACK.
15474
15475  070126  012704  100200                  MOV     #100200,R4      ;EXPECTED FPS.
15476  070132  020405                          CMP     R4,R5           ;IS FPS CORRECT?
15477  070134  001401                          BEQ     2$
15478  070136  104000                          EMT                     ;
15479  070140  012702  000004          2$:     MOV     #4,R2           ;EXPECTED FEC.
15480  070144  020203                          CMP     R2,R3           ;IS FEC CORRECT?
15481  070146  001411                          BEQ     BBBDONE
15482  070150  104000                          EMT                     ;
15483
15484  070152  000000  000000  000000  BBBP1:  .WORD   0,0,0,0
15485  070160  000000
15486  070162  012345  054321  023456  BBBP2:  .WORD   12345,54321,23456,76543
15487  070170  076543
15488
15489
```

```
15490
15491  070172
15492  070172  004767  030002          BBBDONE:
15493                                          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
15494                                                                  ;SEE IF THE USER HAS EXPRESSED
15495                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
15496                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
15497                                                                  ;THE USER TYPED CONTROL G?).
15498
15499
15500                                  ;*****************************************************************************
15501                                  ;TEST 463       DIVF TEST
15502                                  ;*****************************************************************************
15503  070176                          TS463:
15504
15505                                  ;CHECK DIVF WITH (AC=0).
15506  070176  004767  000404          CCC1:   JSR     PC,DIVFSUB
15507  070202  000000  000000          1$:     .WORD   0,0             ;AC
15508  070206  012345  067012          2$:     .WORD   12345,67012     ;FSRC
15509  070212  000000  000000          3$:     .WORD   0,0             ;RES
15510  070216  000000                  4$:     0                       ;FPS BEFORE EXECUTION
15511  070220  000004                          4                       ;FPS AFTER EXECUTION
15512
15513                                  ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE AND IN ROUND MODE.
15514  070222  004737  070606          CCC2:   JSR     PC,@#DIVFSUB
15515  070226  065652  125252          1$:     .WORD   65652,125252    ;AC
15516  070232  065600  000000          2$:     .WORD   65600,0         ;FSRC
15517  070236  040252  125252          3$:     .WORD   40252,125252    ;RES
15518  070242  003000                  4$:     3000                    ;FPS BEFORE EXECUTION.
15519  070244  003000                          3000                    ;FPS AFTER EXECUTION.
15520
15521                                  ;TEST DIVF WITH AC POSITIVE, FSRC POSITIVE.
15522  070246  004767  000334          CCC3:   JSR     PC,DIVFSUB
15523  070252  076400  000000          1$:     .WORD   76400,0         ;AC
15524  070256  076400  000000          2$:     .WORD   76400,0         ;FSRC
15525  070262  040200  000000          3$:     .WORD   40200,0         ;RES
15526  070266  001000                  4$:     1000                    ;FPS BEFORE EXECUTION.
15527  070270  001000                          1000                    ;FPS AFTER EXECUTION.
15528                                  ;TEST DIVF WITH BOTH OPERANDS POSITIVE.
15529  070272  004737  070606          CCC4:   JSR     PC,@#DIVFSUB
15530  070276  056777  177777          1$:     .WORD   56777,177777    ;AC
15531  070302  054200  000000          2$:     .WORD   54200,0         ;FSRC
15532  070306  042777  177777          3$:     .WORD   42777,177777    ;RES
15533  070312  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
15534  070314  000000                          0                       ;FPS AFTER EXECUTION.
15535
15536                                  ;TEST THE DIVF INSTRUCTION:
15537  070316  004737  070606          CCC5:   JSR     PC,@#DIVFSUB
15538  070322  012377  177777          1$:     .WORD   12377,177777    ;AC
15539  070326  012300  000000          2$:     .WORD   12300,0         ;FSRC
15540  070332  040252  125252          3$:     .WORD   40252,125252    ;RES
15541  070336  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
15542  070340  000000                          0                       ;FPS AFTER EXECUTION.
15543
15544                                  ;TEST DIVIDE ALGORITHM. TEST ROUND CONSTANT.
15545  070342  004737  070606          CCC6:   JSR     PC,@#DIVFSUB
```

```
15546  070346  064600  000001          1$:     .WORD   64600,1          ;AC
```

```
15547  070352  066600  000000          2$:     .WORD   66600,0           ;FSRC
15548  070356  036200  000001          3$:     .WORD   36200,1           ;RES
15549  070362  000000                  4$:     0                         ;FPS BEFORE EXECUTION.
15550  070364  000000                          0                         ;FPS AFTER EXECUTION.
15551
15552                                  ;TEST DIVF.
15553  070366  004737  070606          CCC7:   JSR     PC,@#DIVFSUB
15554  070372  034577  177776          1$:     .WORD   34577,177776      ;AC
15555  070376  023400  000000          2$:     .WORD   23400,0           ;FSRC
15556  070402  051377  177776          3$:     .WORD   51377,177776      ;RES
15557  070406  000017                  4$:     17                        ;FPS BEFORE EXECUTION.
15558  070410  000000                          0                         ;FPS AFTER EXECUTION.
15559
15560
15561                                  ;DIVF TEST.
15562  070412  004737  070606          CCC8:   JSR     PC,@#DIVFSUB
15563  070416  067652  125252          1$:     .WORD   67652,125252      ;AC
15564  070422  056500  000000          2$:     .WORD   56500,0           ;FSRC
15565  070426  051343  107070          3$:     .WORD   51343,107070      ;RES
15566  070432  000000                  4$:     0                         ;FPS BEFORE EXECUTION.
15567  070434  000000                          0                         ;FPS AFTER EXECUTION.
15568
15569                                  ;DIVF WITH AC NEGATIVE, FSRC NEGATIVE.
15570  070436  004737  070606          CCC9:   JSR     PC,@#DIVFSUB
15571  070442  140400  000000          1$:     .WORD   140400,0          ;AC
15572  070446  140500  000000          2$:     .WORD   140500,0          ;FSRC
15573  070452  040052  125253          3$:     .WORD   040052,125253     ;RES
15574  070456  000000                  4$:     0                         ;FPS BEFORE EXECUTION.
15575  070460  000000                          0                         ;FPS AFTER EXECUTION.
15576
15577                                  ;DIVF WITH AC NEGATIVE AND FSRC POSITIVE.
15578  070462  004737  070606          CCC10:  JSR     PC,@#DIVFSUB
15579  070466  160077  000000          1$:     .WORD   160077,0          ;AC
15580  070472  040277  000000          2$:     .WORD   40277,0           ;FSRC
15581  070476  160000  000000          3$:     .WORD   160000,0          ;RES
15582  070502  000007                  4$:     7                         ;FPS BEFORE EXECUTION.
15583  070504  000010                          10                        ;FPS AFTER EXECUTION.
15584
15585                                  ;DIVF WITH AC POSITIVE AND FSRC NEGATIVE.
15586  070506  004737  070606          CCC11:  JSR     PC,@#DIVFSUB
15587  070512  040400  000000          1$:     .WORD   40400,0           ;AC
15588  070516  140500  000000          2$:     .WORD   140500,0          ;FSRC
15589  070522  140052  125253          3$:     .WORD   140052,125253     ;RES
15590  070526  000017                  4$:     17                        ;FPS BEFORE EXECUTION.
15591  070530  000010                          10                        ;FPS AFTER EXECUTION.
15592
15593
15594                                  ;TEST DIVF BOTH OPERANDS POSITIVE AND TRUNCATE MODE.
15595  070532  004737  070606          CCC12:  JSR     PC,@#DIVFSUB
15596  070536  060100  000001          1$:     .WORD   60100,1           ;AC
15597  070542  040300  000000          2$:     .WORD   40300,0           ;FSRC
15598  070546  060000  000000          3$:     .WORD   60000,0           ;RES
15599  070552  000052                  4$:     52                        ;FPS BEFORE EXECUTION.
15600  070554  000040                          40                        ;FPS AFTER EXECUTION.
15601
15602                                  ;DIVF WITH POSITIVE OPERANDS AND ROUND MODE.
```

```
15603  070556  004767  000024          CCC13:  JSR     PC,DIVFSUB
15604  070562  060100  000001          1$:     .WORD   60100,1         ;AC
15605  070566  040300  000000          2$:     .WORD   40300,0         ;FSRC
15606  070572  060000  000001          3$:     .WORD   60000,1         ;RES
15607  070576  000005                  4$:     5                       ;FPS BEFORE EXECUTION.
15608  070600  000000                          0                       ;FPS AFTER EXECUTION.
15609
15610  070602  000137  070724                  JMP     @#CCCDONE       ;GO TO NEXT TEST.
15611
15612                                  ;THIS SUBROUTINE, DIVFSUB, IS CALLED TO SET UP, EXECUTE
15613                                  ;AND CHECK THE RESULT OF A DIVF INSTRUCTION. IT IS CALLED THUS:
15614                                  ;
15615                                  ;               JSR     PC,@#DIVFSUB
15616                                  ;       ACARG:  .WORD   X,X             ;AC OPERAND
15617                                  ;       FSRCARG:.WORD   X,X             ;FSRC OPERAND
15618                                  ;       RES:    .WORD   X,X             ;EXPECTED RESULT
15619                                  ;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
15620                                  ;       FPSA:   .WORD   X               ;FPS AFTER EXECUTION
15621                                  ;       ERRES:  .WORD   X,X             ;ERROR RESULT
15622                                  ;       ERR:    ERROR   X               ;RESULT ERROR
15623                                  ;       CONT:                           ;RETURN ADDRESS
15624                                  ;
15625                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
15626                                  ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVF IS EXECUTED.
15627                                  ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
15628                                  ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
15629                                  ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
15630                                  ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
15631                                  ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
15632                                  ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
15633                                  ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
15634                                  ;THEN THE FAILURE IS REPORTED IN DIVFSUB AND CONTROL IS PASSED TO
15635                                  ;CONT. IF NO ERRORS ARE DETECTED THEN DIVFSUB RETURNS CONTROL
15636                                  ;TO CONT.
15637
15638  070606  012601                  DIVFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
15639  070610  012700  000200                  MOV     #200,R0         ;SET FD MODE.
15640  070614  170100                          LDFPS   R0
15641  070616  010100                          MOV     R1,R0           ;LOAD THE AC OPERAND.
15642  070620  172410                          LDD     (R0),AC0
15643  070622  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS
15644  070626  170100                          LDFPS   R0
15645  070630  010100                          MOV     R1,R0
15646  070632  062700  000004                  ADD     #4,R0           ;ESTABLISH A POINTER TO FSRC.
15647
15648  070636  174410                  1$:     DIVF    (R0),AC0        ;TEST INSTRUCTION.
15649
15650  070640  170204                          STFPS   R4              ;GET THE FPS.
15651  070642  012700  000200                  MOV     #200,R0         ;SET FD MODE
15652  070646  170100                          LDFPS   R0
15653
15654  070650  012700  070714                  MOV     #DIVFT,R0       ;GET THE RESULT OF THE DIV
15655  070654  174010                          STD     AC0,(R0)
15656  070656  021061  000010                  CMP     (R0),10(R1)     ;IS THE RESULT CORRECT?
15657  070662  001401                          BEQ     2$
15658  070664  104000                          EMT                     ;
```

```
15659  070666  026061  000002  000012  2$:     CMP     2(R0),12(R1)
15660  070674  001401                          BEQ     3$
15661  070676  104000                          EMT
15662  070700  026104  000016          3$:     CMP     16(R1),R4       ;IS FPS CORRECT?
15663  070704  001401                          BEQ     4$
15664  070706  104000                          EMT
15665  070710  000161  000020          4$:     JMP     20(R1)          ;IF NO ERRORS OCCURRED RETURN.
15666
15667  070714  000000  000000  000000  DIVFT:  .WORD   0,0,0,0
15668  070722  000000
15669
15670  070724                          CCCDONE:
15671  070724  004767  027250                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
15672                                                                  ;SEE IF THE USER HAS EXPRESSED
15673                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
15674                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
15675                                                                  ;THE USER TYPED CONTROL G?).
15676
15677
15678
15679                                   ;*****************************************************************************
15680                                   ;TEST 464       DIVD TEST
15681                                   ;*****************************************************************************
15682  070730                           TS464:
15683
15684                                   ;DIVD TEST WITH POSITIVE OPERANDS AND IN ROUND MODE.
15685  070730  004737  071274           DDD1:   JSR     PC,@#DIVDSUB
15686  070734  034277  000000  000000  1$:     .WORD   34277,0,0,0     ;AC
15687  070742  000000
15688  070744  040277  000000  000000  2$:     .WORD   40277,0,0,0     ;FSRC
15689  070752  000000
15690  070754  034200  000000  000000  3$:     .WORD   34200,0,0,0     ;RES
15691  070762  000000
15692  070764  000200                  4$:     200                     ;FPS BEFORE EXECUTION.
15693  070766  000200                          200                     ;FPS AFTER EXECUTION.
15694
15695                                   ;DIVD WITH AC NEGATIVE AND FSRC POSITIVE IN TRUNCATE MODE.
15696  070770  004737  071274           DDD2:   JSR     PC,@#DIVDSUB
15697  070774  134277  000000  000000  1$:     .WORD   134277,0,0,0    ;AC
15698  071002  000000
15699  071004  040277  000000  000000  2$:     .WORD   40277,0,0,0     ;FSRC
15700  071012  000000
15701  071014  134200  000000  000000  3$:     .WORD   134200,0,0,0    ;RES
15702  071022  000000
15703  071024  000207                  4$:     207                     ;FPS BEFORE EXECUTION.
15704  071026  000210                          210                     ;FPS AFTER EXECUTION.
15705
15706                                   ;DIVD TEST WITH OPERANDS BOTH NEGATIVE AND IN TRUNCATE MODE.
15707  071030  004767  000240           DDD3:   JSR     PC,DIVDSUB
15708  071034  134300  000000  000000  1$:     .WORD   134300,0,0,1    ;AC
15709  071042  000001
15710  071044  140300  000000  000000  2$:     .WORD   140300,0,0,0    ;FSRC
15711  071052  000000
15712  071054  034200  000000  000000  3$:     .WORD   34200,0,0,0     ;RES
15713  071062  000000
15714  071064  000250                  4$:     250                     ;FPS BEFORE EXECUTION.
```

```
15715  071066  000240                          240                     ;FPS AFTER EXECUTION.
15716
15717                                          ;DIVD WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
15718  071070  004737  071274                  DDD4:   JSR     PC,@#DIVDSUB
15719  071074  034300  000000  000000          1$:     .WORD   34300,0,0,1     ;AC
15720  071102  000001
15721  071104  140300  000000  000000          2$:     .WORD   140300,0,0,0    ;FSRC
15722  071112  000000
15723  071114  134200  000000  000000          3$:     .WORD   134200,0,0,1    ;RES
15724  071122  000001
15725  071124  000207                          4$:     207                     ;FPS BEFORE EXECUTION.
15726  071126  000210                                  210                     ;FPS AFTER EXECUTION.
15727
15728                                          ;DIVD TEST.
15729  071130  004737  071274                  DDD5:   JSR     PC,@#DIVDSUB
15730  071134  100400  000000  000000          1$:     .WORD   100400,0,0,0    ;AC
15731  071142  000000
15732  071144  000500  000000  000000          2$:     .WORD   500,0,0,0       ;FSRC
15733  071152  000000
15734  071154  140052  125252                  3$:     .WORD   140052,125252   ;RES
15735  071160  125252  125252                          .WORD   125252,125252
15736  071164  007647                          4$:     7647                    ;FPS BEFORE EXECUTION.
15737  071166  007650                                  7650                    ;FPS AFTER EXECUTION.
15738
15739
15740                                          ;DIVD TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
15741  071170  004737  071274                  DDD6:   JSR     PC,@#DIVDSUB
15742  071174  000400  000000  000000          1$:     .WORD   400,0,0,0       ;AC
15743  071202  000000
15744  071204  100500  000000  000000          2$:     .WORD   100500,0,0,0    ;FSRC
15745  071212  000000
15746  071214  140052  125252                  3$:     .WORD   140052,125252   ;RES
15747  071220  125252  125253                          .WORD   125252,125253
15748  071224  007707                          4$:     7707                    ;FPS BEFORE EXECUTION.
15749  071226  007710                                  7710                    ;FPS AFTER EXECUTION.
15750
15751                                          ;DIVD TEST.
15752  071230  004737  071274                  DDD7:   JSR     PC,@#DIVDSUB
15753  071234  170360  170360                  1$:     .WORD   170360,170360   ;AC
15754  071240  170360  170360                          .WORD   170360,170360
15755  071244  170360  170360                  2$:     .WORD   170360,170360   ;FSRC
15756  071250  170360  170360                          .WORD   170360,170360
15757  071254  040200  000000  000000          3$:     .WORD   40200,0,0,0     ;RES
15758  071262  000000
15759  071264  007717                          4$:     7717                    ;FPS BEFORE EXECUTION.
15760  071266  007700                                  7700                    ;FPS AFTER EXECUTION.
15761
15762  071270  000137  071416                          JMP     @#DDDDONE       ;GO TO NEXT TEST.
15763
15764
15765                                          ;THIS SUBROUTINE, DIVDSUB, IS CALLED TO SET UP, EXECUTE
15766                                          ;AND CHECK THE RESULT OF A DIVD INSTRUCTION. IT IS CALLED THUS:
15767                                          ;
15768                                          ;               JSR     PC,@#DIVDSUB
15769                                          ;       ACARG:  .WORD   X,X,X,X                 ;AC OPERAND
15770                                          ;       FSRCARG:.WORD   X,X,X,X                 ;FSRC OPERAND
```

```
15771                                   ;       RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
15772                                   ;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
15773                                   ;       FPSA:   .WORD   X               ;FPS AFTER EXECUTION
15774                                   ;       ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
15775                                   ;       ERR:    ERROR   X               ;RESULT ERROR
15776                                   ;       CONT:                           ;RETURN ADDRESS
15777                                   ;
15778                                   ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
15779                                   ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, DIVD IS EXECUTED.
15780                                   ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
15781                                   ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
15782                                   ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
15783                                   ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
15784                                   ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
15785                                   ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
15786                                   ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
15787                                   ;THEN THE FAILURE IS REPORTED IN DIVDSUB AND CONTROL IS PASSED TO
15788                                   ;CONT. IF NO ERRORS ARE DETECTED THEN DIVDSUB RETURNS CONTROL
15789                                   ;TO CONT.
15790
15791  071274  012601                   DIVDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
15792  071276  012700  000200                   MOV     #200,R0         ;SET FD MODE.
15793  071302  170100                           LDFPS   R0
15794                  .
15795  071304  010100                           MOV     R1,R0           ;SET UP THE AC0 OPERAND.
15796  071306  172410                           LDD     (R0),AC0
15797  071310  016100  000030                   MOV     30(R1),R0       ;LOAD THE FPS.
15798  071314  170100                           LDFPS   R0
15799
15800  071316  010100                           MOV     R1,R0           ;ESTABLISH A POINTER TO FSRC.
15801  071320  062700  000010                   ADD     #10,R0
15802
15803  071324  174410                   1$:     DIVD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
15804  071326  170204                           STFPS   R4              ;GET THE FPS.
15805  071330  012700  000200                   MOV     #200,R0         ;SET FD MODE.
15806  071334  170100                           LDFPS   R0
15807  071336  012700  071406                   MOV     #DIVDT,R0       ;GET THE RESULT.
15808  071342  174010                           STD     AC0,(R0)
15809  071344  010102                           MOV     R1,R2           ;CHECK THE RESULT.
15810  071346  062702  000020                   ADD     #20,R2
15811  071352  012703  071406                   MOV     #DIVDT,R3
15812  071356  012705  000004                   MOV     #4,R5
15813  071362  022223                   2$:     CMP     (R2)+,(R3)+
15814  071364  001401                           BEQ     3$
15815  071366  104000                           EMT                     ;
15816  071370  077504                   3$:     SOB     R5,2$
15817
15818  071372  026104  000032                   CMP     32(R1),R4       ;IS FPS CORRECT?
15819  071376  001401                           BEQ     4$
15820  071400  104000                           EMT                     ;
15821  071402  000161  000034           4$:     JMP     34(R1)          ;RETURN.
15822  071406  000000  000000  000000   DIVDT:  .WORD   0,0,0,0
15823  071414  000000
15824
15825  071416                           DDDDONE:
15826  071416  004767  026556                   JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
```

```
15827                                           ;SEE IF THE USER HAS EXPRESSED
15828                                           ;THE DESIRE TO CHANGE THE SOFTWARE
15829                                           ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
15830                                           ;THE USER TYPED CONTROL G?).
15831
15832
15833
15834                                   ;*******************************************************************************
15835                                   ;TEST 465       MULF TEST
15836                                   ;*******************************************************************************
15837  071422                           TS465:
15838
15839                                   ;MULF WITH (FSRC=AC=0)
15840  071422  004737  072032           EEE1:   JSR     PC,@#MULFSUB
15841  071426  000000  000000           1$:     .WORD   0,0             ;AC
15842  071432  000000  000000           2$:     .WORD   0,0             ;FSRC
15843  071436  000000  000000           3$:     .WORD   0,0             ;RES
15844  071442  007517                   4$:     7517                    ;FPS BEFORE EXECUTION.
15845  071444  007504                           7504                    ;FPS AFTER EXECUTION.
15846
15847                                   ;MULF WITH (FSRC=0).
15848  071446  004737  072032           EEE2:   JSR     PC,@#MULFSUB
15849  071452  071625  034435           1$:     .WORD   71625,34435     ;AC
15850  071456  000000  000000           2$:     .WORD   0,0             ;FSRC
15851  071462  000000  000000           3$:     .WORD   0,0             ;RES
15852  071466  000013                   4$:     13                      ;FPS BEFORE EXECUTION.
15853  071470  000004                           4                       ;FPS AFTER EXECUTION.
15854
15855                                   ;MULF WITH (AC=0)
15856  071472  004737  072032           EEE3:   JSR     PC,@#MULFSUB
15857  071476  000000  000000           1$:     .WORD   0,0             ;AC
15858  071502  071625  153443           2$:     .WORD   071625,153443   ;FSRC
15859  071506  000000  000000           3$:     .WORD   0,0             ;RES
15860  071512  007500                   4$:     7500                    ;FPS BEFORE EXECUTION.
15861  071514  007504                           7504                    ;FPS AFTER EXECUTION.
15862
15863                                   ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
15864  071516  004737  072032           EEE4:   JSR     PC,@#MULFSUB
15865  071522  040200  000000           1$:     .WORD   40200,0         ;AC
15866  071526  040177  177777           2$:     .WORD   40177,-1        ;FSRC
15867  071532  040177  177777           3$:     .WORD   40177,-1        ;RES
15868  071536  000017                   4$:     17                      ;FPS BEFORE EXECUTION.
15869  071540  000000                           0                       ;FPS AFTER EXECUTION.
15870
15871                                   ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN TRUNCATE MODE.
15872  071542  004767  000264           EEE5:   JSR     PC,MULFSUB
15873  071546  040177  177777           1$:     .WORD   40177,-1        ;AC
15874  071552  040200  000000           2$:     .WORD   40200,0         ;FSRC
15875  071556  040177  177777           3$:     .WORD   40177,-1        ;RES
15876  071562  000040                   4$:     40                      ;FPS BEFORE EXECUTION.
15877  071564  000040                           40                      ;FPS AFTER EXECUTION.
15878
15879                                   ;MULF WITH BOTH OPERANDS POSITIVE NORMALIZE TEST.
15880  071566  004737  072032           EEE6:   JSR     PC,@#MULFSUB
15881  071572  040100  000000           1$:     .WORD   40100,0         ;AC
15882  071576  040100  000000           2$:     .WORD   40100,0         ;FSRC
```

```
15883  071602  040020  000000  3$:     .WORD   40020,0          ;RES
15884  071606  000012          4$:     12                      ;FPS BEFORE EXECUTION.
15885  071610  000000                  0                       ;FPS AFTER EXECUTION.
15886
15887                          ;MULF WITH BOTH OPERANDS POSITIVE IN ROUND MODE.
15888  071612  004737  072032  EEE7:   JSR     PC,@#MULFSUB
15889  071616  017500  000000  1$:     .WORD   17500,0         ;AC
15890  071622  023652  125252  2$:     .WORD   23652,125252    ;FSRC
15891  071626  003177  177777  3$:     .WORD   3177,-1         ;RES
15892  071632  007417          4$:     7417                    ;FPS BEFORE EXECUTION.
15893  071634  007400                  7400                    ;FPS AFTER EXECUTION.
15894
15895                          ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
15896  071636  004737  072032  EEE8:   JSR     PC,@#MULFSUB
15897  071642  040342  000000  1$:     .WORD   40342,0         ;AC
15898  071646  176542  000000  2$:     .WORD   176542,0        ;FSRC
15899  071652  176707  102000  3$:     .WORD   176707,102000   ;RES
15900  071656  000007          4$:     7                       ;FPS BEFORE EXECUTION.
15901  071660  000010                  10                      ;FPS AFTER EXECUTION.
15902
15903                          ;MULF WITH AC NEGATIVE AND FSRC POSITIVE IN ROUND MODE.
15904  071662  004737  072032  EEE9:   JSR     PC,@#MULFSUB
15905  071666  140200  000000  1$:     .WORD   140200,0        ;AC
15906  071672  007417  007417  2$:     .WORD   7417,7417       ;FSRC
15907  071676  107417  007417  3$:     .WORD   107417,7417     ;RES
15908  071702  000000          4$:     0                       ;FPS BEFORE EXECUTION.
15909  071704  000010                  10                      ;FPS AFTER EXECUTION.
15910
15911                          ;MULF WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
15912  071706  004737  072032  EEE10:  JSR     PC,@#MULFSUB
15913  071712  144600  000000  1$:     .WORD   144600,0        ;AC
15914  071716  154000  000000  2$:     .WORD   154000,0        ;FSRC
15915  071722  060400  000000  3$:     .WORD   60400,0         ;RES
15916  071726  000017          4$:     17                      ;FPS BEFORE EXECUTION.
15917  071730  000000                  0                       ;FPS AFTER EXECUTION.
15918
15919                          ;MULF BOTH OPERANDS NEGATIVE IN ROUND MODE.
15920  071732  004737  072032  EEE11:  JSR     PC,@#MULFSUB
15921  071736  140300  000000  1$:     .WORD   140300,0        ;AC
15922  071742  160000  000001  2$:     .WORD   160000,1        ;FSRC
15923  071746  060100  000002  3$:     .WORD   60100,2         ;RES
15924  071752  000010          4$:     10                      ;FPS BEFORE EXECUTION.
15925  071754  000000                  0                       ;FPS AFTER EXECUTION.
15926
15927                          ;MULF WITH AC POSITIVE AND FSRC NEGATIVE IN TRUNCATE MODE.
15928  071756  004737  072032  EEE12:  JSR     PC,@#MULFSUB
15929  071762  060000  000001  1$:     .WORD   60000,1         ;AC
15930  071766  140300  000000  2$:     .WORD   140300,0        ;FSRC
15931  071772  160100  000001  3$:     .WORD   160100,1        ;RES
15932  071776  007547          4$:     7547                    ;FPS BEFORE EXECUTION.
15933  072000  007550                  7550                    ;FPS AFTER EXECUTION.
15934
15935                          ;MULF WITH AC POSITIVE AND FSRC POSITIVE IN ROUND MODE.
15936  072002  004737  072032  EEE13:  JSR     PC,@#MULFSUB
15937  072006  040277  000000  1$:     .WORD   40277,0         ;AC
15938  072012  060000  000001  2$:     .WORD   60000,1         ;FSRC
```

```
15939  072016  060077  000001          3$:     .WORD   60077,1          ;RES
15940  072022  000014                  4$:     14                      ;FPS BEFORE EXECUTION.
15941  072024  000000                          0                       ;FPS AFTER EXECUTION.
15942
15943  072026  000167  000116                  JMP     EEEDONE         ;GO TO THE NEXT TEST.
15944
15945                                  ;THIS SUBROUTINE, MULFSUB, IS CALLED TO SET UP, EXECUTE
15946                                  ;AND CHECK THE RESULT OF A MULF INSTRUCTION. IT IS CALLED THUS:
15947                                  ;
15948                                  ;               JSR     PC,@#MULFSUB
15949                                  ;       ACARG:  .WORD   X,X             ;AC OPERAND
15950                                  ;       FSRCARG:.WORD   X,X             ;FSRC OPERAND
15951                                  ;       RES:    .WORD   X,X             ;EXPECTED RESULT
15952                                  ;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
15953                                  ;       FPSA:   .WORD   X               ;FPS AFTER EXECUTION
15954                                  ;       ERRES:  .WORD   X,X             ;ERROR RESULT
15955                                  ;       ERR:    ERROR   X               ;RESULT ERROR
15956                                  ;       CONT:                           ;RETURN ADDRESS
15957                                  ;
15958                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
15959                                  ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULF IS EXECUTED.
15960                                  ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
15961                                  ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
15962                                  ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
15963                                  ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
15964                                  ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
15965                                  ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
15966                                  ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
15967                                  ;THEN THE FAILURE IS REPORTED IN MULFSUB AND CONTROL IS PASSED TO
15968                                  ;CONT. IF NO ERRORS ARE DETECTED THEN MULFSUB RETURNS CONTROL
15969                                  ;TO CONT.
15970
15971  072032  012601                  MULFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
15972  072034  012700  000200                  MOV     #200,R0         ;SET FD MODE.
15973  072040  170100                          LDFPS   R0
15974  072042  010100                          MOV     R1,R0           ;LOAD THE AC OPERAND.
15975  072044  172410                          LDD     (R0),AC0
15976  072046  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS
15977  072052  170100                          LDFPS   R0
15978  072054  010100                          MOV     R1,R0
15979  072056  062700  000004                  ADD     #4,R0           ;ESTABLISH A POINTER TO FSRC.
15980
15981  072062  171010                  1$:     MULF    (R0),AC0        ;TEST INSTRUCTION.
15982
15983  072064  170204                          STFPS   R4              ;GET THE FPS.
15984  072066  012700  000200                  MOV     #200,R0         ;SET FD MODE
15985  072072  170100                          LDFPS   R0
15986
15987  072074  012700  072140                  MOV     #MULFT,R0       ;GET THE RESULT OF THE MULF.
15988  072100  174010                          STD     AC0,(R0)
15989  072102  021061  000010                  CMP     (R0),10(R1)     ;IS THE RESULT CORRECT?
15990  072106  001401                          BEQ     2$
15991  072110  104000                          EMT                     ;
15992  072112  026061  000002  000012  2$:     CMP     2(R0),12(R1)
15993  072120  001401                          BEQ     3$
15994  072122  104000                          EMT                     ;
```

```
15995  072124  026104  000016          3$:     CMP     16(R1),R4       ;IS FPS CORRECT?
15996  072130  001401                          BEQ     4$
15997  072132  104000                          EMT                     ;
15998  072134  000161  000020          4$:     JMP     20(R1)          ;IF NO ERRORS OCCURRED RETURN.
15999
16000  072140  000000  000000  000000  MULFT:  .WORD   0,0,0,0
16001  072146  000000
16002
16003  072150                          EEEDONE:
16004  072150  004767  026024                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
16005                                                                  ;SEE IF THE USER HAS EXPRESSED
16006                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
16007                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
16008                                                                  ;THE USER TYPED CONTROL G?).
16009
16010
16011
16012                                   ;*****************************************************************************
16013                                   ;TEST 466       MULD TEST
16014                                   ;*****************************************************************************
16015  072154                           TS466:
16016
16017                                   ;MULD TEST WITH AC POSITIVE AND FSRC POSITIVE.
16018  072154  004737  072360           FFF1:   JSR     PC,@#MULDSUB
16019  072160  040200  000000  000000   1$:     .WORD   40200,0,0,0     ;AC
16020  072166  000000
16021  072170  023777  177777  177777   2$:     .WORD   23777,-1,-1,-1  ;FSRC
16022  072176  177777
16023  072200  023777  177777  177777   3$:     .WORD   23777,-1,-1,-1  ;RES
16024  072206  177777
16025  072210  000217                   4$:     217                     ;FPS BEFORE EXECUTION.
16026  072212  000200                           200                     ;FPS AFTER ECECUTION.
16027
16028                                   ;MULD TEST WITH BOTH OPERANDS POSITIVE TRUNCATION TEST.
16029  072214  004767  000140           FFF2:   JSR     PC,MULDSUB
16030  072220  065400  000000  000000   1$:     .WORD   65400,0,0,1     ;AC
16031  072226  000001
16032  072230  037577  177777  177777   2$:     .WORD   37577,-1,-1,-2  ;FSRC
16033  072236  177776
16034  072240  064777  177777  177777   3$:     .WORD   64777,-1,-1,-1  ;RES
16035  072246  177777
16036  072250  000247                   4$:     247                     ;FPS BEFORE EXECUTION.
16037  072252  000240                           240                     ;FPS AFTER EXECUTION.
16038
16039                                   ;MULD TEST WITH BOTH OPERANDS NEGATIVE IN ROUND MODE.
16040  072254  004737  072360           FFF3:   JSR     PC,@#MULDSUB
16041  072260  137577  177777  177777   1$:     .WORD   137577,-1,-1,-2 ;AC
16042  072266  177776
16043  072270  165400  000000  000000   2$:     .WORD   165400,0,0,1    ;FSRC
16044  072276  000001
16045  072300  065000  000000  000000   3$:     .WORD   65000,0,0,0     ;RES
16046  072306  000000
16047  072310  007717                   4$:     7717                    ;FPS BEFORE EXECUTION.
16048  072312  007700                           7700                    ;FPS AFTER EXECUTION.
16049
16050                                   ;MULD   TEST WITH AC POSITIVE AND FSRC NEGATIVE IN ROUND MODE.
```

```
16051  072314  004737  072360          FFF4:   JSR     PC,@#MULDSUB
16052  072320  017500  000000  000000  1$:     .WORD   17500,0,0,0     ;AC
16053  072326  000000
16054  072330  123652  125252          2$:     .WORD   123652,125252   ;FSRC
16055  072334  125252  125252                  .WORD   125252,125252
16056  072340  103177  177777  177777  3$:     .WORD   103177,-1,-1,-1 ;RES
16057  072346  177777
16058  072350  000200                  4$:     200                     ;FPS BEFORE EXECUTION.
16059  072352  000210                          210                     ;FPS AFTER EXECUTION.
16060
16061  072354  000167  000122                  JMP     FFFDONE
16062
16063                                  ;THIS SUBROUTINE, MULDSUB, IS CALLED TO SET UP, EXECUTE
16064                                  ;AND CHECK THE RESULT OF A MULD INSTRUCTION. IT IS CALLED THUS:;
16065                                  ;               JSR     PC,@#MULDSUB
16066                                  ;       ACARG:  .WORD   X,X,X,X                 ;AC OPERAND
16067                                  ;       FSRCARG:.WORD   X,X,X,X                 ;FSRC OPERAND
16068                                  ;       RES:    .WORD   X,X,X,X                 ;EXPECTED RESULT
16069                                  ;       FPSB:   .WORD   X                       ;FPS BEFORE EXECUTION
16070                                  ;       FPSA:   .WORD   X                       ;FPS AFTER EXECUTION
16071                                  ;       ERRES:  .WORD   X,X,X,X                 ;ERROR RESULT
16072                                  ;       ERR:    ERROR   X                       ;RESULT ERROR
16073                                  ;       CONT:                                   ;RETURN ADDRESS
16074                                  ;
16075                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC OPERAND). THEN
16076                                  ;FPSB IS LOADED INTO THE FPS. THE INSTRUCTION, MULD IS EXECUTED.
16077                                  ;AFTER THE EXECUTION THE RESULT IS CHECKED AGAINST THE
16078                                  ;EXPECTED CORRECT RESULT, RES. IF IT IS CORRECT THEN THE FPS
16079                                  ;IS CHECKED WITH THE EXPECTED CORRECT FPS, FPSA. IF THE FPS WAS
16080                                  ;INCORRECT THEN IT IS REPORTED. IF THE RESULT WAS INCORRECT IT
16081                                  ;IS COMPARED WITH ERRES IN AN ATTEMPT TO ANALYSE THE ERROR. IF
16082                                  ;THE INCORRECT RESULT MATCHED ERRES THEN CONTROL IS PASSED TO
16083                                  ;THE ERROR CALL AT ERR. IF THE INCORRECT RESULT DID NOT MATCH ERRES
16084                                  ;THEN THE FAILURE IS REPORTED IN MULDSUB AND CONTROL IS PASSED TO
16085                                  ;CONT. IF NO ERRORS ARE DETECTED THEN MULDSUB RETURNS CONTROL
16086                                  ;TO CONT.
16087
16088  072360  012601                  MULDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
16089  072362  012700  000200                  MOV     #200,R0         ;SET FD MODE.
16090  072366  170100                          LDFPS   R0
16091
16092  072370  010100                          MOV     R1,R0           ;SET UP THE AC0 OPERAND.
16093  072372  172410                          LDD     (R0),AC0
16094  072374  016100  000030                  MOV     30(R1),R0       ;LOAD THE FPS.
16095  072400  170100                          LDFPS   R0
16096
16097  072402  010100                          MOV     R1,R0           ;ESTABLISH A POINTER TO FSRC.
16098  072404  062700  000010                  ADD     #10,R0
16099
16100  072410  171010                  1$:     MULD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
16101
16102  072412  170204                          STFPS   R4              ;GET THE FPS.
16103  072414  012700  000200                  MOV     #200,R0         ;SET FD MODE.
16104  072420  170100                          LDFPS   R0
16105
16106  072422  012700  072472                  MOV     #MULDT,R0       ;GET THE RESULT.
```

```
16107  072426  174010                        STD     AC0,(R0)
16108  072430  010102                        MOV     R1,R2           ;CHECK THE RESULT.
16109  072432  062702  000020                ADD     #20,R2
16110  072436  012703  072472                MOV     #MULDT,R3
16111  072442  012705  000004                MOV     #4,R5
16112  072446  022223                2$:     CMP     (R2)+,(R3)+
16113  072450  001401                        BEQ     3$
16114  072452  104000                        EMT                     ;
16115  072454  077504                3$:     SOB     R5,2$
16116
16117  072456  026104  000032                CMP     32(R1),R4       ;IS FPS CORRECT?
16118  072462  001401                        BEQ     4$
16119  072464  104000                        EMT                     ;
16120  072466  000161  000034        4$:     JMP     34(R1)          ;RETURN.
16121
16122  072472  000000  000000  000000  MULDT:  .WORD   0,0,0,0
16123  072500  000000
16124  072502                        FFFDONE:
16125  072502  004767  025472                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
16126                                                                ;SEE IF THE USER HAS EXPRESSED
16127                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
16128                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
16129                                                                ;THE USER TYPED CONTROL G?).
16130
16131
16132
16133                                ;*****************************************************************************
16134                                ;TEST 467       UNDER\OVER FLOW, USING MULF WITH TRAPS DISABLED, TEST
16135                                ;*****************************************************************************
16136  072506                        TS467:
16137
16138                                ;UNDERFLOW, WITH EXPONENT OF RESULT = -129
16139  072506  004737  072652        III1:   JSR     PC,@#OVUNFNT
16140  072512  020200  000000        1$:     .WORD   20200,0         ;AC
16141  072516  020000  000000        2$:     .WORD   20000,0         ;FSRC
16142  072522  000000  000000        3$:     .WORD   0,0             ;RES
16143  072526  000000                5$:     0                       ;FPS BEFORE EXECUTION.
16144  072530  000004                        4                       ;FPS AFTER EXECUTION.
16145  072532  000012                6$:     12                      ;FEC
16146  072534  177777                        -1                      ;FLAG
16147
16148                                ;UNDERFLOW, WITH EXPONENT OF RESULT = -193
16149  072536  004737  072652        III2:   JSR     PC,@#OVUNFNT
16150  072542  010200  000000        1$:     .WORD   10200,0         ;AC
16151  072546  010000  000000        2$:     .WORD   10000,0         ;FSRC
16152  072552  000000  000000        3$:     .WORD   0,0             ;RES
16153  072556  005013                5$:     5013                    ;FPS BEFORE EXECUTION.
16154  072560  005004                        5004                    ;FPS AFTER EXECUTION.
16155  072562  000012                6$:     12                      ;FEC
16156  072564  177777                        -1                      ;FLAG
16157                                ;OVERFLOW, EXPONENT OF RESULT = 128
16158  072566  004737  072652        III3:   JSR     PC,@#OVUNFNT
16159  072572  060200  000000        1$:     .WORD   60200,0         ;AC
16160  072576  060000  000000        2$:     .WORD   60000,0         ;FSRC
16161  072602  000000  000000        3$:     .WORD   0,0             ;RES
16162  072606  000000                5$:     0                       ;FPS BEFORE EXECUTION.
```

```
16163  072610  000006                    6                       ;FPS AFTER EXECUTION.
16164  072612  000010           6$:      10                      ;FEC
16165  072614  000000                    0                       ;FLAG
16166                           ;OVERFLOW, EXPONENT OF RESULT = 130
16167  072616  004737  072652   III4:    JSR     PC,@#OVUNFNT
16168  072622  060200  000000   1$:      .WORD   60200,0         ;AC
16169  072626  060200  000000   2$:      .WORD   60200,0         ;FSRC
16170  072632  000000  000000   3$:      .WORD   0,0             ;RES
16171  072636  006011           5$:      6011                    ;FPS BEFORE EXECUTION.
16172  072640  006006                    6006                    ;FPS AFTER EXECUTION.
16173  072642  000010           6$:      10                      ;FEC
16174  072644  000000                    0                       ;FLAG
16175  072646  000167  000132   8$:      JMP     IIIDONE         ;GO TO NEXT TEST.
16176
16177
16178                           ;THIS SUBROUTINE, OVUNFNT, IS USED TO SET UP THE OPERANDS, EXECUTE
16179                           ;THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
16180                           ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
16181                           ;TO IT IS MADE THUS:
16182                           ;
16183                           ;               ACARG:   .WORD   X,X             ;AC OPERAND
16184                           ;               FSRCARG:.WORD    X,X             ;FSRC OPERAND
16185                           ;               RES:     .WORD   X,X             ;EXPECTED RESULT
16186                           ;               ERRES:   .WORD   X,X             ;ERROR RESULT
16187                           ;               FPSB:    .WORD   X               ;FPS BEFORE EXECUTION
16188                           ;               FPSA:    .WORD   X               ;FPS AFTER EXECUTION
16189                           ;               FEC:     .WORD   X               ;EXPECTED FEC
16190                           ;               FLAG:    .WORD   X               ;0/-1,OVER/UNDER FLOW FLAG
16191                           ;               ERR1:    ERROR   X               ;TRAP ERROR.
16192                           ;               BR       CONT
16193                           ;               ERR2:    ERROR   X               ;DATA, RESULT ERROR
16194                           ;               CONT:                            ;RETURN ADDRESS
16195                           ;
16196
16197                           ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
16198                           ;THE MULF INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE
16199                           ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
16200                           ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFNT RETURNS CONTROL
16201                           ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFNT
16202                           ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
16203                           ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
16204                           ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
16205                           ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFNT
16206                           ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
16207                           ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFNT WILL
16208                           ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
16209                           ;IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNFNT WILL READ THE FEC.
16210                           ;SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNFNT WILL
16211                           ;STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
16212                           ;FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNFNT WILL REPORT
16213                           ;THE ERROR AND RETURN TO CONT. NOTE THAT OVUNFNT USES THE FLAG
16214                           ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
16215                           ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
16216
16217  072652  012601           OVUNFNT:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
16218  072654  012700  000200           MOV     #200,R0         ;SET FD MODE.
```

```
16219  072660  170100                          LDFPS   R0
16220
16221  072662  010100                          MOV     R1,R0           ;LOAD AC0, OPERAND.
16222  072664  172410                          LDD     (R0),AC0
16223  072666  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS
16224  072672  170100                          LDFPS   R0
16225  072674  012737  072754  000244          MOV     #25$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
16226                                                                  ;OF ERROR.
16227  072702  010100                          MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
16228  072704  062700  000004                  ADD.    #4,R0
16229
16230  072710  171010                  1$:     MULF    (R0),AC0        ;TEST INSTRUCTION.
16231
16232  072712  170204                  2$:     STFPS   R4              ;GET FPS.
16233  072714  170305                          STST    R5              ;GET FEC.
16234  072716  012700  000200                  MOV     #200,R0         ;SET FD MODE.
16235  072722  170100                          LDFPS   R0
16236  072724  012700  072774                  MOV     #OVFNTT,R0      ;GET THE RESULT.
16237  072730  174010                          STD     AC0,(R0)
16238  072732  012700  072774                  MOV     #OVFNTT,R0      ;CHECK THE RESULT.
16239  072736  010102                          MOV     R1,R2
16240  072740  062702  000010                  ADD     #10,R2
16241  072744  012703  000002                  MOV     #2,R3
16242  072750  022022                  3$:     CMP     (R0)+,(R2)+
16243  072752  001401                          BEQ     5$
16244  072754                          25$:
16245  072754  104000                          EMT                     ;
16246  072756  077304                  5$:     SOB     R3,3$
16247
16248  072760  026104  000016                  CMP     16(R1),R4       ;WAS FPS CORRECT?
16249  072764  001401                          BEQ     4$
16250  072766  104000                          EMT                     ;
16251  072770  000161  000024          4$:     JMP     24(R1)          ;RETURN, TEST COMPLETED.
16252
16253  072774  000000  000000  000000  OVFNTT: .WORD   0,0,0,0
16254  073002  000000
16255
16256  073004                          IIIDONE:
16257  073004  004767  025170                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
16258                                                                  ;SEE IF THE USER HAS EXPRESSED
16259                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
16260                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
16261                                                                  ;THE USER TYPED CONTROL G?).
16262
16263
16264
16265
16266                                   ;*****************************************************************************
16267                                   ;TEST 470       UNDER\OVER FLOW, USING MULD WITH TRAP DISABLED, TEST
16268                                   ;*****************************************************************************
16269  073010                          TS470:
16270
16271                                   ;UNDERFLOW, EXPONENT OF RESULT=-129
16272  073010  004737  073234          JJJ1:   JSR     PC,@#OVUNDNT
16273  073014  020200  000000          1$:     .WORD   20200,0         ;AC
16274  073020  127272  000000                  .WORD   127272,0
```

```
16275  073024  020000  000000  000000  2$:     .WORD   20000,0,0,0      ;FSRC
16276  073032  000000
16277  073034  000000  000000  000000  3$:     .WORD   0,0,0,0          ;RES
16278  073042  000000
16279  073044  000200                  5$:     200                      ;FPS BEFORE EXECUTION.
16280  073046  000204                          204                      ;FPS AFTER EXECUTION.
16281  073050  000012                  6$:     12                       ;FEC
16282  073052  177777                          -1                       ;FLAG
16283
16284                                  ;UNDERFLOW, EXPONENT OF RESULT = -193
16285  073054  004737  073234          JJJ2:   JSR     PC,@#OVUNDNT
16286  073060  010200  000000          1$:     .WORD   10200,0          ;AC
16287  073064  123456  000000                  .WORD   123456,0
16288  073070  010000  000000  000000  2$:     .WORD   10000,0,0,0      ;FSRC
16289  073076  000000
16290  073100  000000  000000  000000  3$:     .WORD   0,0,0,0          ;RES
16291  073106  000000
16292  073110  005213                  5$:     5213                     ;FPS BEFORE EXECUTION.
16293  073112  005204                          5204                     ;FPS AFTER EXECUTION.
16294  073114  000012                  6$:     12                       ;FEC
16295  073116  177777                          -1                       ;FLAG
16296
16297                                  ;OVERFLOW, EXPONENT OF RESULT = 128
16298  073120  004737  073234          JJJ3:   JSR     PC,@#OVUNDNT
16299  073124  060200  000000          1$:     .WORD   60200,0          ;AC
16300  073130  065432  000000                  .WORD   65432,0
16301  073134  060000  000000  000000  2$:     .WORD   60000,0,0,0      ;FSRC
16302  073142  000000
16303  073144  000000  000000  000000  3$:     .WORD   0,0,0,0          ;RES
16304  073152  000000
16305  073154  000200                  5$:     200                      ;FPS BEFORE EXECUTION.
16306  073156  000206                          206                      ;FPS AFTER EXECUTION.
16307  073160  000010                  6$:     10                       ;FEC
16308  073162  000000                          0                        ;FLAG
16309
16310                                  ;OVERFLOW, EXPONENT OF RESULT = 130
16311  073164  004737  073234          JJJ4:   JSR     PC,@#OVUNDNT
16312  073170  060200  000000          1$:     .WORD   60200,0          ;AC
16313  073174  125252  000000                  .WORD   125252,0
16314  073200  060200  000000  000000  2$:     .WORD   60200,0,0,0      ;FSRC
16315  073206  000000
16316  073210  000000  000000  000000  3$:     .WORD   0,0,0,0          ;RES
16317  073216  000000
16318  073220  006211                  5$:     6211                     ;FPS BEFORE EXECUTION.
16319  073222  006206                          6206                     ;FPS AFTER EXECUTION.
16320  073224  000010                  6$:     10                       ;FEC
16321  073226  000000                          0                        ;FLAG
16322  073230  000137  073366          8$:     JMP     @#JJJDONE        ;GO TO NEXT TEST
16323
16324                                  ;THIS SUBROUTINE, OVUNDNT, IS USED TO SET UP THE OPERANDS, EXECUTE
16325                                  ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
16326                                  ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
16327                                  ;TO IT IS MADE THUS:
16328                                  ;
16329                                  ;               ACARG:  .WORD   X,X,X,X          ;AC OPERAND
16330                                  ;               FSRCARG:.WORD   X,X,X,X          ;FSRC OPERAND
```

```
16331                                   ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
16332                                   ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
16333                                   ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
16334                                   ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
16335                                   ;               FEC:    .WORD   X               ;EXPECTED FEC
16336                                   ;               FLAG:   .WORD   X               ;0/-1,OVER/UNDER FLOW FLAG
16337                                   ;               ERR1:   ERROR   X               ;TRAP ERROR.
16338                                   ;               BR      CONT
16339                                   ;               ERR2:   ERROR   X               ;DATA, RESULT ERROR
16340                                   ;               CONT:                           ;RETURN ADDRESS
16341                                   ;
16342                                   ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
16343                                   ;THE MULD INSTRUCTION IS EXECUTED. IF NO TRAP OCCURS THEN THE
16344                                   ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
16345                                   ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDNT RETURNS CONTROL
16346                                   ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDNT
16347                                   ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
16348                                   ;MULD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
16349                                   ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
16350                                   ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDNT
16351                                   ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
16352                                   ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDNT WILL
16353                                   ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
16354                                   ;IF A TRAP OCCURS (IT SHOULD NOT) THEN OVUNDNT WILL READ THE FEC.
16355                                   ;SHOULD THE FEC MATCH THE ANTICIPATED FEC OVUNDNT WILL
16356                                   ;STORE ALL DATA AND TRANSFER CONTROL TO THE ERROR CALL AT ERR1. IF THE
16357                                   ;FEC IS NOT THE SAME AS THE ANTICIPATED FEC OVUNDNT WILL REPORT
16358                                   ;THE ERROR AND RETURN TO CONT. NOTE THAT OVUNDNT USES THE FLAG
16359                                   ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
16360                                   ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
16361
16362 073234 012601                    OVUNDNT:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
16363 073236 012700 000200                      MOV     #200,R0         ;SET FD MODE.
16364 073242 170100                             LDFPS   R0
16365
16366 073244 010100                             MOV     R1,R0           ;LOAD AC0, OPERAND.
16367 073246 172410                             LDD     (R0),AC0
16368 073250 016100 000030                      MOV     30(R1),R0       ;LOAD THE FPS.
16369 073254 170100                             LDFPS   R0
16370 073256 012737 073336 000244               MOV     #25$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
16371                                                                   ;OF ERROR.
16372 073264 010100                             MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
16373 073266 062700 000010                      ADD     #10,R0
16374
16375 073272 171010                     1$:     MULD    (R0),AC0        ;TEST INSTRUCTION.
16376
16377 073274 170204                     2$:     STFPS   R4              ;GET FPS.
16378 073276 170305                             STST    R5              ;GET FEC.
16379 073300 012700 000200                      MOV     #200,R0         ;SET FD MODE.
16380 073304 170100                             LDFPS   R0
16381 073306 012700 073356                      MOV     #OVDNTT,R0      ;GET THE RESULT.
16382 073312 174010                             STD     AC0,(R0)
16383 073314 012700 073356                      MOV     #OVDNTT,R0      ;CHECK THE RESULT.
16384 073320 010102                             MOV     R1,R2
16385 073322 062702 000020                      ADD     #20,R2
16386 073326 012703 000004                      MOV     #4,R3
```

```
16387  073332  022022                  3$:     CMP     (R0)+,(R2)+
16388  073334  001401                          BEQ     5$
16389  073336                          25$:
16390  073336  104000                          EMT                     ;
16391  073340  077304                  5$:     SOB     R3,3$
16392
16393  073342  026104  000032                  CMP     32(R1),R4       ;WAS FPS CORRECT?
16394  073346  001401                          BEQ     4$
16395  073350  104000                          EMT                     ;
16396  073352  000161  000040          4$:     JMP     40(R1)          ;RETURN, TEST COMPLETED.
16397
16398  073356  000000  000000  000000  OVDNTT: .WORD   0,0,0,0
16399  073364  000000
16400
16401  073366                          JJJDONE:
16402  073366  004767  024606                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
16403                                                                  ;SEE IF THE USER HAS EXPRESSED
16404                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
16405                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
16406                                                                  ;THE USER TYPED CONTROL G?).
16407
16408
16409
16410
16411                                   ;*****************************************************************************
16412                                   ;TEST 471       UNDER\OVER FLOW, USING MULF WITH TRAPS ENABLED, TEST
16413                                   ;*****************************************************************************
16414  073372                           TS471:
16415
16416                                   ;UNDERFLOW, EXPONENT OF RESULT = -129
16417  073372  004737  073536           KKK1:   JSR     PC,@#OVUNFT
16418  073376  020123  045676           1$:     .WORD   20123,45676     ;AC
16419  073402  020200  000000           2$:     .WORD   20200,0         ;FSRC
16420  073406  000123  045676           3$:     .WORD   123,45676       ;RES
16421  073412  002000                   5$:     2000                    ;FPS BEFORE EXECUTION.
16422  073414  102004                           102004                  ;FPS AFTER EXECUTION.
16423  073416  000012                   6$:     12                      ;FEC
16424  073420  177777                           -1                      ;FLAG
16425
16426                                   ;UNDERFLOW, EXPONENT OF THE RESULT = -193
16427  073422  004737  073536           KKK3:   JSR     PC,@#OVUNFT
16428  073426  010127  127272           1$:     .WORD   10127,127272    ;AC
16429  073432  010200  000000           2$:     .WORD   10200,0         ;FSRC
16430  073436  060127  127272           3$:     .WORD   60127,127272    ;RES
16431  073442  007017                   5$:     7017                    ;FPS BEFORE EXECUTION.
16432  073444  107000                           107000                  ;FPS AFTER EXECUTION.
16433  073446  000012                   6$:     12                      ;FEC
16434  073450  177777                           -1
16435
16436                                   ;OVERFLOW, EXPONENT OF THE RESULT = 128
16437  073452  004737  073536           KKK4:   JSR     PC,@#OVUNFT
16438  073456  060252  125252           1$:     .WORD   60252,125252    ;AC
16439  073462  060000  000000           2$:     .WORD   60000,0         ;FSRC
16440  073466  000052  125252           3$:     .WORD   000052,125252   ;RES
16441  073472  001000                   5$:     1000                    ;FPS BEFORE EXECUTION.
16442  073474  101006                           101006                  ;FPS AFTER EXECUTION.
```

```
16443  073476  000010                  6$:     10                      ;FEC
16444  073500  000000                          0                       ;FLAG
16445
16446                                  ;OVERFLOW, EXPONENT OF RESULT = 130
16447  073502  004737  073536          KKK5:   JSR     PC,@#OVUNFT
16448  073506  060345  067654          1$:     .WORD   60345,67654     ;AC
16449  073512  060200  000000          2$:     .WORD   60200,0         ;FSRC
16450  073516  000345  067654          3$:     .WORD   345,67654       ;RES
16451  073522  007015                  5$:     7015                    ;FPS BEFORE EXECUTION.
16452  073524  107002                          107002                  ;FPS AFTER EXECUTION.
16453  073526  000010                  6$:     10                      ;FEC
16454  073530  000000                          0                       ;FLAG
16455  073532  000167  000162          8$:     JMP     KKKDONE
16456
16457                                  ;THIS SUBROUTINE, OVUNFT, IS USED TO SET UP THE OPERANDS, EXECUTE
16458                                  ;THE MULF INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
16459                                  ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
16460                                  ;TO IT IS MADE THUS:
16461                                  ;
16462                                  ;               ACARG:  .WORD   X,X             ;AC OPERAND
16463                                  ;               FSRCARG:.WORD   X,X             ;FSRC OPERAND
16464                                  ;               RES:    .WORD   X,X             ;EXPECTED RESULT
16465                                  ;               ERRES:  .WORD   X,X             ;ERROR RESULT
16466                                  ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
16467                                  ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
16468                                  ;               FEC:    .WORD   X               ;EXPECTED FEC
16469                                  ;               FLAG:   .WORD   X               ;0/-1,OVER/UNDER FLOW FLAG
16470                                  ;               ERR1:   ERROR   X               ;TRAP ERROR.
16471                                  ;               BR      CONT
16472                                  ;               ERR2:   ERROR   X               ;DATA, RESULT ERROR
16473                                  ;               CONT:                           ;RETURN ADDRESS
16474                                  ;
16475                                  ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
16476                                  ;THE MULF INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
16477                                  ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
16478                                  ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNFT RETURNS CONTROL
16479                                  ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNFT
16480                                  ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
16481                                  ;IN THE SAME WAY. IF THE RESULT OF THE
16482                                  ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
16483                                  ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
16484                                  ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNFT
16485                                  ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
16486                                  ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNFT WILL
16487                                  ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
16488                                  ;IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
16489                                  ;NOTE THAT OVUNFNT USES THE FLAG
16490                                  ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
16491                                  ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
16492
16493  073536  012601                  OVUNFT: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
16494  073540  012700  000200                  MOV     #200,R0         ;SET FD MODE.
16495  073544  170100                          LDFPS   R0
16496  073546  010100                          MOV     R1,R0           ;LOAD AC0, OPERAND.
16497  073550  172410                          LDD     (R0),AC0
16498  073552  016100  000014                  MOV     14(R1),R0       ;LOAD THE FPS.
```

```
16499  073556  170100                          LDFPS   R0
16500  073560  012737  073602  000244          MOV     #50$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
                                                                        ;OF ERROR.
16501
16502  073566  010100                          MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
16503  073570  062700  000004                  ADD     #4,R0
16504
16505  073574  171010                  1$:     MULF    (R0),AC0        ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
16506  073576  170000                  2$:     CFCC
16507  073600  104000                          EMT                     ;
16508  073602  011602                  50$:    MOV     (SP),R2         ;TRAP TO HERE AND SEE IF THE PC OF THE
16509  073604  020227  073576                  CMP     R2,#2$          ;TRAP WAS THAT OF THE MULF INSTRUCTION.
16510  073610  001401                          BEQ     51$
16511  073612  104000                          EMT                     ;
16512  073614  022626                  51$:    CMP     (SP)+,(SP)+     ;RESET THE STACK
16513  073616  170204                          STFPS   R4              ;GET FPS.
16514  073620  170305                          STST    R5              ;GET FEC.
16515  073622  012700  000200                  MOV     #200,R0         ;SET FD MODE.
16516  073626  170100                          LDFPS   R0
16517  073630  012700  073710                  MOV     #OVFTT,R0       ;GET THE RESULT.
16518  073634  174010                          STD     AC0,(R0)
16519  073636  012700  073710                  MOV     #OVFTT,R0       ;CHECK THE RESULT.
16520  073642  010102                          MOV     R1,R2
16521  073644  062702  000010                  ADD     #10,R2
16522  073650  012703  000002                  MOV     #2,R3
16523  073654  022022                  3$:     CMP     (R0)+,(R2)+
16524  073656  001401                          BEQ     5$
16525  073660  104000                          EMT                     ;
16526  073662  077304                  5$:     SOB     R3,3$
16527
16528  073664  026104  000016                  CMP     16(R1),R4       ;WAS FPS CORRECT?
16529  073670  001401                          BEQ     6$
16530  073672  104000                          EMT                     ;
16531  073674  026105  000020          6$:     CMP     20(R1),R5       ;IS FEC CORRECT?
16532  073700  001401                          BEQ     4$
16533  073702  104000                          EMT                     ;
16534  073704  000161  000024          4$:     JMP     24(R1)          ;RETURN, TEST COMPLETED.
16535
16536  073710  000000  000000  000000  OVFTT:  .WORD   0,0,0,0
16537  073716  000000
16538
16539  073720                          KKKDONE:
16540  073720  004767  024254                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
16541                                                                  ;SEE IF THE USER HAS EXPRESSED
16542                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
16543                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
16544                                                                  ;THE USER TYPED CONTROL G?).
16545
16546
16547
16548
16549                           ;*****************************************************************************
16550                           ;TEST 472       UNDER\OVER FLOW, USING MULD WITH TRAPS ENABLED, TEST
16551                           ;*****************************************************************************
16552  073724                  TS472:
16553
16554                           ;UNDERFLOW, EXPONENT OF RESULT = -129
```

```
16555  073724  004737  074150          LLL1:   JSR     PC,@#OVUNDT
16556  073730  020052  125252          1$:     .WORD   20052,125252    ;AC
16557  073734  125252  125252                  .WORD   125252,125252
16558  073740  020300  000000  000000  2$:     .WORD   20300,0,0,0     ;FSRC
16559  073746  000000
16560  073750  000177  177777  177777  3$:     .WORD   177,-1,-1,-1    ;RES
16561  073756  177777
16562  073760  002200                  5$:     2200                    ;FPS BEFORE EXECUTION.
16563  073762  102204                          102204                  ;FPS AFTER EXECUTION.
16564  073764  000012                  6$:     12                      ;FEC
16565  073766  177777                          -1                      ;FLAG
16566
16567                                  ;UNDERFLOW, EXPONENT OF THE RESULT = -193
16568  073770  004737  074150          LLL2:   JSR     PC,@#OVUNDT
16569  073774  010327  127272          1$:     .WORD   10327,127272    ;AC
16570  074000  036363  045454                  .WORD   36363,45454
16571  074004  010000  000000  000000  2$:     .WORD   10000,0,0,0     ;FSRC
16572  074012  000000
16573  074014  060127  127272          3$:     .WORD   60127,127272    ;RES
16574  074020  036363  045454                  .WORD   36363,45454
16575  074024  007217                  5$:     7217                    ;FPS BEFORE EXECUTION.
16576  074026  107200                          107200                  ;FPS AFTER EXECUTION.
16577  074030  000012                  6$:     12                      ;FEC
16578  074032  177777                          -1                      ;FLAG
16579
16580                                  ;OVERFLOW, EXPONENT OF THE RESULT = 128
16581  074034  004737  074150          LLL3:   JSR     PC,@#OVUNDT
16582  074040  060252  125252          1$:     .WORD   60252,125252    ;AC
16583  074044  125252  125252                  .WORD   125252,125252   ;FSRC
16584  074050  160100  000000  000000  2$:     .WORD   160100,0,0,0    ;FSRC
16585  074056  000000
16586  074060  100177  177777  177777  3$:     .WORD   100177,-1,-1,-1 ;RES
16587  074066  177777
16588  074070  001200                  5$:     1200                    ;FPS BEFORE EXECUTION.
16589  074072  101216                          101216                  ;FPS AFTER EXECUTION.
16590  074074  000010                  6$:     10                      ;FEC
16591  074076  000000                          0                       ;FLAG
16592
16593                                  ;OVERFLOW, EXPONENT OF THE RESULT = 130
16594  074100  004737  074150          LLL4:   JSR     PC,@#OVUNDT
16595  074104  060345  067654          1$:     .WORD   60345,67654     ;AC
16596  074110  056765  045676                  .WORD   56765,45676
16597  074114  060200  000000  000000  2$:     .WORD   60200,0,0,0     ;FSRC
16598  074122  000000
16599  074124  000345  067654          3$:     .WORD   345,67654       ;RES
16600  074130  056765  045676                  .WORD   56765,45676
16601  074134  007215                  5$:     7215                    ;FPS BEFORE EXECUTION.
16602  074136  107202                          107202                  ;FPS AFTER EXECUTION.
16603  074140  000010                  6$:     10                      ;FEC
16604  074142  000000                          0                       ;FLAG
16605  074144  000137  074332          8$:     JMP     @#LLLDONE
16606
16607                                  ;THIS SUBROUTINE, OVUNDT, IS USED TO SET UP THE OPERANDS, EXECUTE
16608                                  ;THE MULD INSTRUCTION AND CHECK THE RESULTS OF AN INSTRUCTION WITH
16609                                  ;OPERANDS WHICH SHOULD RESULT IN EITHER OVERFLOW OR UNDERFLOW. A CALL
16610                                  ;TO IT IS MADE THUS:
```

```
16611                   ;
16612                   ;       ACARG:   .WORD   X,X,X,X        ;AC OPERAND
16613                   ;       FSRCARG:.WORD    X,X,X,X        ;FSRC OPERAND
16614                   ;       RES:     .WORD   X,X,X,X        ;EXPECTED RESULT
16615                   ;       ERRES:   .WORD   X,X,X,X        ;ERROR RESULT
16616                   ;       FPSB:    .WORD   X              ;FPS BEFORE EXECUTION
```

```
16617                                   ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
16618                                   ;               FEC:    .WORD   X               ;EXPECTED FEC
16619                                   ;               FLAG:   .WORD   X               ;0/-1,OVER/UNDER FLOW FLAG
16620                                   ;               ERR1:   ERROR   X               ;TRAP ERROR.
16621                                   ;               BR      CONT
16622                                   ;               ERR2:   ERROR   X               ;DATA, RESULT ERROR
16623                                   ;               CONT:                           ;RETURN ADDRESS
16624                                   ;
16625                                   ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
16626                                   ;THE MULD INSTRUCTION IS EXECUTED. IF THE TRAP OCCURS THEN THE
16627                                   ;RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
16628                                   ;COMPARED WITH FPSA IF THIS TOO IS CORRECT OVUNDT RETURNS CONTROL
16629                                   ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD OVUNDT
16630                                   ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. THE FEC IS TREATED
16631                                   ;IN THE SAME WAY. IF THE RESULT OF THE
16632                                   ;MULF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
16633                                   ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
16634                                   ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN OVUNDT
16635                                   ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR2. OTHERWISE THE
16636                                   ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND OVUNDT WILL
16637                                   ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
16638                                   ;IF NO TRAP OCCURS CONTROL IS PASSED TO ERR1.
16639                                   ;NOTE THAT OVUNDNT USES THE FLAG
16640                                   ;TO TELL WHETHER OR NOT THESE PARTICULAR OPERANDS WILL RESULT IN
16641                                   ;UNDERFLOW (FLAG=-1) OR OVERFLOW (FLAG=0).
16642
16643  074150  012601                   OVUNDT: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
16644  074152  012700  000200                   MOV     #200,R0         ;SET FD MODE.
16645  074156  170100                           LDFPS   R0
16646
16647  074160  010100                           MOV     R1,R0           ;LOAD AC0, OPERAND.
16648  074162  172410                           LDD     (R0),AC0
16649  074164  016100  000030                   MOV     30(R1),R0       ;LOAD THE FPS.
16650  074170  170100                           LDFPS   R0
16651  074172  012737  074214  000244           MOV     #50$,@#FPVECT   ;SET UP THE FP TRAP VECTOR IN CASE
16652                                                                   ;OF ERROR.
16653  074200  010100                           MOV     R1,R0           ;COMPUTE THE ADDRESS OF FSRC.
16654  074202  062700  000010                   ADD     #10,R0
16655
16656  074206  171010                   1$:     MULD    (R0),AC0        ;TEST INSTRUCTION. SHOULD CAUSE TRAP.
16657  074210  170000                   2$:     CFCC
16658  074212  104000                           EMT                     ;
16659  074214  011602                   50$:    MOV     (SP),R2         ;TRAP TO HERE AND SEE IF THE PC OF THE
16660  074216  020227  074210                   CMP     R2,#2$          ;TRAP WAS THAT OF THE MULF INSTRUCTION.
16661  074222  001401                           BEQ     51$             ;BRANCH IF YES.
16662  074224  104000                           EMT                     ;
16663  074226  022626                   51$:    CMP     (SP)+,(SP)+     ;RESET THE STACK
16664  074230  170204                           STFPS   R4              ;GET FPS.
16665  074232  170305                           STST    R5              ;GET FEC.
16666  074234  012700  000200                   MOV     #200,R0         ;SET FD MODE.
16667  074240  170100                           LDFPS   R0
16668  074242  012700  074322                   MOV     #OVDTT,R0       ;GET THE RESULT.
16669  074246  174010                           STD     AC0,(R0)
16670  074250  012700  074322                   MOV     #OVDTT,R0       ;CHECK THE RESULT.
16671  074254  010102                           MOV     R1,R2
16672  074256  062702  000020                   ADD     #20,R2
```

```
16673  074262  012703  000004            MOV     #4,R3
16674  074266  022022                3$:  CMP     (R0)+,(R2)+
16675  074270  001401                     BEQ     5$
16676  074272  104000                     EMT                     ;
16677  074274  077304                5$:  SOB     R3,3$
16678  074276  026104  000032             CMP     32(R1),R4       ;WAS FPS CORRECT?
16679  074302  001401                     BEQ     6$
16680  074304  104000                     EMT                     ;
16681  074306  026105  000034        6$:  CMP     34(R1),R5       ;IS FEC CORRECT?
16682  074312  001401                     BEQ     4$
16683  074314  104000                     EMT                     ;
16684  074316  000161  000040        4$:  JMP     40(R1)          ;RETURN, TEST COMPLETED.
16685
16686  074322  000000  000000  000000 OVDTT: .WORD  0,0,0,0
16687  074330  000000
16688
16689  074332                     LLLDONE:
16690  074332  004767  023642             JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
16691                                                             ;SEE IF THE USER HAS EXPRESSED
16692                                                             ;THE DESIRE TO CHANGE THE SOFTWARE
16693                                                             ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
16694                                                             ;THE USER TYPED CONTROL G?).
16695
16696
16697
16698
16699
16700                              ;*********************************************************************************************
16701                              ;TEST 473       MODF TEST
16702                              ;*********************************************************************************************
16703  074336                     TS473:
16704
16705                              ;MODF WITH (FSRC=AC=0)
16706  074336  004737  075062     GGG1:   JSR     PC,@#MODFSUB
16707  074342  000000  000000     1$:     .WORD   0,0             ;AC
16708  074346  000000  000000     2$:     .WORD   0,0             ;FSRC
16709  074352  000000  000000     3$:     .WORD   0,0             ;FRACTIONAL RES.
16710  074356  000000  000000     4$:     .WORD   0,0             ;INTEGER RES.
16711  074362  000013             7$:     13                      ;FPS BEFORE EXECUTION.
16712  074364  000004                     4                       ;FPS AFTER EXECUTION.
16713
16714                              ;MODF TEST, WITH (FSRC=0)
16715  074366  004737  075062     GGG2:   JSR     PC,@#MODFSUB
16716  074372  123456  076543     1$:     .WORD   123456,76543    ;AC
16717  074376  000000  000000     2$:     .WORD   0,0             ;FSRC
16718  074402  000000  000000     3$:     .WORD   0,0             ;FRACTIONAL RES.
16719  074406  000000  000000     4$:     .WORD   0,0             ;INTEGER RESULT.
16720  074412  000000             7$:     0                       ;FPS BEFORE EXECUTION.
16721  074414  000004                     4                       ;FPS AFTER EXECUTION.
16722
16723                              ;MODF TEST WITH (AC=0)
16724  074416  004737  075062     GGG3:   JSR     PC,@#MODFSUB
16725  074422  000000  000000     1$:     .WORD   0,0             ;AC
16726  074426  076543  021234     2$:     .WORD   76543,21234     ;FSRC
16727  074432  000000  000000     3$:     .WORD   0,0             ;FRACTIONAL RES.
16728  074436  000000  000000     4$:     .WORD   0,0             ;INTEGER RES.
```

```
16729  074442  000003
16730  074444  000004
16731
16732
16733  074446  004737  075062
16734  074452  046252  125252
16735  074456  040300  000000
16736  074462  000000  000000
16737  074466  046377  177777
16738  074472  000013
16739  074474  000004
16740
16741
16742  074476  004737  075062
16743  074502  077652  125252
16744  074506  040300  000000
16745  074512  000000  000000
16746  074516  077777  177777
16747  074522  000000
16748  074524  000004
16749
16750
16751  074526  004737  075062
16752  074532  046200  000001
16753  074536  040340  000000
16754  074542  000000  000000
16755  074546  046340  000001
16756  074552  000013
16757  074554  000004
16758
16759
16760  074556  004737  075062
16761  074562  046000  000001
16762  074566  040340  000000
16763  074572  040100  000000
16764  074576  046140  000001
16765  074602  000000
16766  074604  000000
16767
16768
16769  074606  004737  075062
16770  074612  042577  177777
16771  074616  040200  000000
16772  074622  040177  176000
16773  074626  042577  140000
16774  074632  000000
16775  074634  000000
16776
16777
16778  074636  004737  075062
16779  074642  042577  140001
16780  074646  040200  000000
16781  074652  034600  000000
16782  074656  042577  140000
16783  074662  000000
16784  074664  000000
```

```
7$:     3                       ;FPS BEFORE EXECUTION.
        4                       ;FPS AFTER EXECUTION.

;MODF TEST WITH EXPONENT OF THE RESULT = 25
GGG4:   JSR     PC,@#MODFSUB
1$:     .WORD   46252,125252    ;AC
2$:     .WORD   40300,0         ;FSRC
3$:     .WORD   0,0             ;FRACTIONAL RES.
4$:     .WORD   46377,-1        ;INTEGER RES.
7$:     13                      ;FPS BEFORE EXECUTION.
        4                       ;FPS AFTER EXECUTION.

;MODF TEST WITH EXPONENT OF THE RESULT = 127
GGG5:   JSR     PC,@#MODFSUB
1$:     .WORD   77652,125252    ;AC
2$:     .WORD   40300,0         ;FSRC
3$:     .WORD   0,0             ;FRACTIONAL RES.
4$:     .WORD   77777,-1        ;INTEGER RES.
7$:     0                       ;FPS BEFORE EXECUTION.
        4                       ;FPS AFTER EXECUTION.

;MODF TEST WITH EXPONENT OF RESULT = 25
GGG6:   JSR     PC,@#MODFSUB
1$:     .WORD   46200,1         ;AC
2$:     .WORD   40340,0         ;FSRC
3$:     .WORD   0,0             ;FRACTIONAL RES.
4$:     .WORD   46340,1         ;INTEGER RES.
7$:     13                      ;FPS BEFORE EXECUTION.
        4                       ;FPS AFTER EXECUTION.

;MODF   TEST WITH EXPONENT OF THE RESULT = 24
GGG7:   JSR     PC,@#MODFSUB
1$:     .WORD   46000,1         ;AC
2$:     .WORD   40340,0         ;FSRC
3$:     .WORD   40100,0         ;FRACTIONAL RES.
4$:     .WORD   46140,1         ;INTEGER RESULT.
7$:     0                       ;FPS BEFORE EXECUTION.
        0                       ;FPS AFTER EXECUTION.

;MODF TEST WITH EXPONENT OF THE RESULT = 10
GGG8:   JSR     PC,@#MODFSUB
1$:     .WORD   42577,-1        ;AC
2$:     .WORD   40200,0         ;FSRC
3$:     .WORD   40177,176000    ;FRACTIONAL RES.
4$:     .WORD   42577,140000    ;INTEGER RES.
7$:     0                       ;FPS BEFORE EXECUTION.
        0                       ;FPS AFTER EXECUTION.

;MODF TEST WITH THE EXPONENT OF THE RESULT = 10
GGG9:   JSR     PC,@#MODFSUB
1$:     .WORD   42577,140001    ;AC
2$:     .WORD   40200,0         ;FSRC
3$:     .WORD   34600,0         ;FRACTIONAL RES.
4$:     .WORD   42577,140000    ;INTEGER RES.
7$:     0                       ;FPS BEFORE EXECUTION.
        0                       ;FPS AFTER EXECUTION.
```

```
16785
16786
16787  074666  004737  075062  ;MODF TEST WITH EXPONENT OF THE RESULT = 9
16788  074672  042377  100000  GGG10:  JSR     PC,@#MODFSUB
16789  074676  040200  000000  1$:     .WORD   42377,100000    ;AC
16790  074702  000000  000000  2$:     .WORD   40200,0         ;FSRC
16791  074706  042377  100000  3$:     .WORD   0,0             ;FRACTIONAL RES.
16792  074712  000013          4$:     .WORD   42377,100000    ;INTEGER RES.
16793  074714  000004          7$:     13                      ;FPS BEFORE EXECUTION.
16794                                  4                       ;FPS AFTER EXECUTION.
16795
16796  074716  004737  075062  ;MODF TEST WITH EXPONENT OF THE RESULT = 0
16797  074722  040177  177777  GGG11:  JSR     PC,@#MODFSUB
16798  074726  040200  000000  1$:     .WORD   40177,-1        ;AC
16799  074732  040177  177777  2$:     .WORD   40200,0         ;FSRC
16800  074736  000000  000000  3$:     .WORD   40177,-1        ;FRACTIONAL RES.
16801  074742  000017          4$:     .WORD   0,0             ;INTEGER RES.
16802  074744  000000          7$:     17                      ;FPS BEFORE EXECUTION.
16803                                  0                       ;FPS AFTER EXECUTION.
16804
16805  074746  004737  075062  ;MODF TEST WITH EXPONENT OF THE RESULT = -15
16806  074752  034377  177777  GGG12:  JSR     PC,@#MODFSUB
16807  074756  040200  000000  1$:     .WORD   34377,-1        ;AC
16808  074762  034377  177777  2$:     .WORD   40200,0         ;FSRC
16809  074766  000000  000000  3$:     .WORD   34377,-1        ;FRACTIONAL RES.
16810  074772  000000          4$:     .WORD   0,0             ;INTEGER RES.
16811  074774  000000          7$:     0                       ;FPS BEFORE EXECUTION.
16812                                  0                       ;FPS AFTER EXECUTION.
16813
16814  074776  004737  075062  ;MODF TEST WITH EXPONENT OF RESULT = -64, IN ROUND MODE
16815  075002  020000  000001  GGG13:  JSR     PC,@#MODFSUB
16816  075006  040300  000000  1$:     .WORD   20000,1         ;AC
16817  075012  020100  000002  2$:     .WORD   40300,0         ;FSRC
16818  075016  000000  000000  3$:     .WORD   20100,2         ;FRACTIONAL RES.
16819  075022  000000          4$:     .WORD   0,0             ;INTEGER RES.
16820  075024  000000          7$:     0                       ;FPS BEFORE EXECUTION.
16821                                  0                       ;FPS AFTER EXECUTION.
16822
16823  075026  004737  075062  ;MODF TEST WITH EXPONENT OF RESULT = 11
16824  075032  142777  170000  GGG14:  JSR     PC,@#MODFSUB
16825  075036  040200  000000  1$:     .WORD   142777,170000   ;AC
16826  075042  140000  000000  2$:     .WORD   40200,0         ;FSRC
16827  075046  142777  160000  3$:     .WORD   140000,0        ;FRACTIONAL RES.
16828  075052  000007          4$:     .WORD   142777,160000   ;INTEGER RES.
16829  075054  000010          7$:     7                       ;FPS BEFORE EXECUTION.
16830  075056  000167  000204          10                      ;FPS AFTER EXECUTION.
16831                          9$:     JMP     GGGDONE         ;GO TO NEXT TEST.
16832
16833                          ;THIS SUBROUTINE, MODFSUB, IS CALLED TO SETUP THE
16834                          ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
16835                          ;IT IS CALLED THUS:
16836                          ;
16837                          ;       ACARG:   .WORD   X,X            ;AC OPERAND
16838                          ;       FSRCARG: .WORD   X,X            ;FSRC OPERAND
16839                          ;       FRES:    .WORD   X,X            ;FRACTIONAL RESULT
16840                          ;       INTRES:  .WORD   X,X            ;INTEGER RESULT
```

```
16841                                   ;               ERFRES: .WORD   X,X             ;ERROR FRACTION RESULT
16842                                   ;               ERINTRES:.WORD  X,X             ;ERROR INTEGER RESULT
16843                                   ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
16844                                   ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
16845                                   ;               ERR1:   ERROR   X               ;FRACTION ERROR
16846                                   ;                       BR      CONT
16847                                   ;               ERR2:   ERROR   X               ;INTEGER ERROR
16848                                   ;               CONT:                           ;RETURN ADDRESS
16849                                   ;
16850                                   ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODF
16851                                   ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
16852                                   ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
16853                                   ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
16854                                   ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
16855                                   ;THEN MODFSUB WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
16856                                   ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
16857                                   ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
16858                                   ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
16859                                   ;FAILURE MATCHES THE TRUE RESULT THEN  MODFSUB PASSES CONTROL TO THE
16860                                   ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
16861                                   ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
16862                                   ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
16863                                   ;IF A MATCH IS MADE HOWEVER, MODFSUB WILL RETURN CONTROL TO THE ERROR
16864                                   ;CALL AT ERR2.
16865
16866  075062  012601                   MODFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
16867  075064  012700  000200                   MOV     #200,R0         ;SET FD MODE.
16868  075070  170100                           LDFPS   R0
16869  075072  010100                           MOV     R1,R0           ;SET UP AC0
16870  075074  172410                           LDD     (R0),AC0
16871  075076  012700  075256                   MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
16872  075102  172510                           LDD     (R0),AC1
16873  075104  016100  000020                   MOV     20(R1),R0       ;SET UP THE FPS.
16874  075110  170100                           LDFPS   R0
16875  075112  010100                           MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
16876  075114  062700  000004                   ADD     #4,R0
16877
16878  075120  171410                   1$:     MODF    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
16879
16880  075122  170204                           STFPS   R4              ;GET THE FPS.
16881  075124  012700  000200                   MOV     #200,R0         ;SET FD MODE.
16882  075130  170100                           LDFPS   R0
16883  075132  012700  075236                   MOV     #MODFT0,R0      ;GET THE FRACTIONAL RESULT.
16884  075136  174010                           STD     AC0,(R0)
16885  075140  012700  075246                   MOV     #MODFT1,R0      ;GET THE INTEGER RESULT.
16886  075144  174110                           STD     AC1,(R0)
16887  075146  012702  075236                   MOV     #MODFT0,R2      ;CHECK THE FRACTIONAL RESULT.
16888  075152  026112  000010                   CMP     10(R1),(R2)
16889  075156  001401                           BEQ     2$
16890  075160  104000                           EMT                     ;
16891  075162  026162  000012  000002  2$:     CMP     12(R1),2(R2)
16892  075170  001401                           BEQ     3$
16893  075172  104000                           EMT                     ;
16894  075174  012702  075246           3$:     MOV     #MODFT1,R2      ;CHECK THE INTEGER RESULT.
16895  075200  026112  000014                   CMP     14(R1),(R2)
16896  075204  001401                           BEQ     4$
```

```
16897  075206  104000                          EMT                     ;
16898  075210  026162  000016  000002  4$:     CMP     16(R1),2(R2)
16899  075216  001401                          BEQ     5$
16900  075220  104000                          EMT                     ;
16901  075222  026104  000022          5$:     CMP     22(R1),R4       ;CHECK THE FPS.
16902  075226  001401                          BEQ     9$
16903  075230  104000                          EMT                     ;
16904  075232  000161  000024          9$:     JMP     24(R1)          ;RETURN.
16905
16906  075236  000000  000000  000000  MODFT0: .WORD   0,0,0,0
16907  075244  000000
16908
16909  075246  000000  000000  000000  MODFT1: .WORD   0,0,0,0
16910  075254  000000
16911
16912  075256  177777  177777  177777  MODP1:  .WORD   -1,-1,-1,-1
16913  075264  177777
16914
16915  075266                          GGGDONE:
16916  075266  004767  022706                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
16917                                                                  ;SEE IF THE USER HAS EXPRESSED
16918                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
16919                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
16920                                                                  ;THE USER TYPED CONTROL G?).
16921
16922
16923
16924
16925                                   ;*****************************************************************************************
16926                                   ;TEST 474       MODD TEST
16927                                   ;*****************************************************************************************
16928  075272                          TS474:
16929
16930                                   ;MODD WITH (FSRC=AC=0)
16931  075272  004737  076306          HHH1:   JSR     PC,@#MODDSUB
16932  075276  000000  000000  000000  1$:     .WORD   0,0,0,0         ;AC
16933  075304  000000
16934  075306  000000  000000  000000  2$:     .WORD   0,0,0,0         ;FSRC
16935  075314  000000
16936  075316  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
16937  075324  000000
16938  075326  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
16939  075334  000000
16940  075336  000200                  7$:     200                     ;FPS BEFORE EXECUTION.
16941  075340  000204                          204                     ;FPS AFTER EXECUTION.
16942
16943                                   ;MODD TEST WITH FSRC=0
16944  075342  004737  076306          HHH2:   JSR     PC,@#MODDSUB
16945  075346  012345  067012          1$:     .WORD   012345,67012    ;AC
16946  075352  034567  012345                  .WORD   34567,012345
16947  075356  000000  000000  000000  2$:     .WORD   0,0,0,0         ;FSRC
16948  075364  000000
16949  075366  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
16950  075374  000000
16951  075376  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
16952  075404  000000
```

```
16953  075406  000213                          7$:     213                     ;FPS BEFORE EXECUTION.
16954  075410  000204                                  204                     ;FPS AFTER EXECUTION.
16955
16956                                          ;MODD TEST WITH (AC=0)
16957  075412  004737  076306                  HHH3:   JSR     PC,@#MODDSUB
16958  075416  000000  000000  000000          1$:     .WORD   0,0,0,0         ;AC
16959  075424  000000
16960  075426  072727  127272                  2$:     .WORD   72727,127272    ;FSRC
16961  075432  072727  127272                          .WORD   72727,127272
16962  075436  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
16963  075444  000000
16964  075446  000000  000000  000000          4$:     .WORD   0,0,0,0         ;INTEGER RES.
16965  075454  000000
16966  075456  000213                          7$:     213                     ;FPS BEFORE EXECUTION.
16967  075460  000204                                  204                     ;FPS AFTER EXECUTION.
16968
16969                                          ;MODD TEST WITH EXPONENT OF THE RESULT = 57
16970  075462  004737  076306                  HHH4:   JSR     PC,@#MODDSUB
16971  075466  056252  125252                  1$:     .WORD   56252,125252    ;AC
16972  075472  125252  125250                          .WORD   125252,125250
16973  075476  040300  000000  000000          2$:     .WORD   40300,0,0,0     ;FSRC
16974  075504  000000
16975  075506  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
16976  075514  000000
16977  075516  056377  177777  177777          4$:     .WORD   56377,-1,-1,-4  ;INTEGER RES.
16978  075524  177774
16979  075526  000213                          7$:     213                     ;FPS BEFORE EXECUTION.
16980  075530  000204                                  204                     ;FPS AFTER EXECUTION.
16981
16982                                          ;MODD TEST WITH EXPONENT OF THE RESULT = 79
16983  075532  004737  076306                  HHH5:   JSR     PC,@#MODDSUB
16984  075536  140240  000000  000000          1$:     .WORD   140240,0,0,0    ;AC
16985  075544  000000
16986  075546  063714  146314                  2$:     .WORD   63714,146314    ;FSRC
16987  075552  133572  167737                          .WORD   133572,167737
16988  075556  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
16989  075564  000000
16990  075566  163777  177777                  4$:     .WORD   163777,-1       ;INTEGER RES.
16991  075572  162531  125726                          .WORD   162531,125726
16992  075576  000210                          7$:     210                     ;FPS BEFORE EXECUTION.
16993  075600  000204                                  204                     ;FPS AFTER EXECUTION.
16994
16995                                          ;MODD TEST WITH EXPONENT OF THE RESULT = 57
16996  075602  004737  076306                  HHH6:   JSR     PC,@#MODDSUB
16997  075606  056200  000000  000000          1$:     .WORD   56200,0,0,1     ;AC
16998  075614  000001
16999  075616  040340  000000  000000          2$:     .WORD   40340,0,0,0     ;FSRC
17000  075624  000000
17001  075626  000000  000000  000000          3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
17002  075634  000000
17003  075636  056340  000000  000000          4$:     .WORD   56340,0,0,1     ;INTEGER RES.
17004  075644  000001
17005  075646  000213                          7$:     213                     ;FPS BEFORE EXECUTION.
17006  075650  000204                                  204                     ;FPS AFTER EXECUTION.
17007
17008                                          ;MODD TEST WITH EXPONENT OF THE RESULT = 56
```

```
17009 075652 004737 076306          HHH7:   JSR     PC,@#MODDSUB
17010 075656 056000 000000 000000   1$:     .WORD   56000,0,0,1     ;AC
17011 075664 000001
17012 075666 040340 000000 000000   2$:     .WORD   40340,0,0,0     ;FSRC
17013 075674 000000
17014 075676 040100 000000 000000   3$:     .WORD   40100,0,0,0     ;FRACTIONAL RES.
17015 075704 000000
17016 075706 056140 000000 000000   4$:     .WORD   56140,0,0,1     ;INTEGER RES.
17017 075714 000001
17018 075716 000213                 7$:     213                     ;FPS BEFORE EXECUTION.
17019 075720 000200                         200                     ;FPS AFTER EXECUTION.
17020
17021                                       ;MODD TEST WITH EXPONENT OF THE RESULT = 36
17022 075722 004737 076306          HHH8:   JSR     PC,@#MODDSUB
17023 075726 051177 177777 177777   1$:     .WORD   51177,-1,-1,-1  ;AC
17024 075734 177777
17025 075736 040200 000000 000000   2$:     .WORD   40200,0,0,0     ;FSRC
17026 075744 000000
17027 075746 040177 177760 000000   3$:     .WORD   40177,-20,0,0   ;FRACTIONAL RES.
17028 075754 000000
17029 075756 051177 177777 177760   4$:     .WORD   51177,-1,-20,0  ;INTEGER RES.
17030 075764 000000
17031 075766 000217                 7$:     217                     ;FPS BEFORE EXECUTION.
17032 075770 000200                         200                     ;FPS AFTER EXECUTION.
17033
17034                                       ;MODD TEST WITH EXPONENT OF THE RESULT = 30
17035 075772 004737 076306          HHH9:   JSR     PC,@#MODDSUB
17036 075776 040200 000000 000000   1$:     .WORD   40200,0,0,0     ;AC
17037 076004 000000
17038 076006 047577 177777          2$:     .WORD   47577,-1        ;FSRC
17039 076012 176000 000001                  .WORD   176000,1
17040 076016 031600 000000 000000   3$:     .WORD   31600,0,0,0     ;FRACTIONAL RES.
17041 076024 000000
17042 076026 047577 177777          4$:     .WORD   47577,-1        ;INTEGER RES.
17043 076032 176000 000000                  .WORD   176000,0
17044 076036 000200                 7$:     200                     ;FPS BEFORE EXECUTION.
17045 076040 000200                         200                     ;FPS AFTER EXECUTION.
17046
17047                                       ;MODD TEST WITH EXPONENT OF THE RESULT = 31
17048 076042 004737 076306          HHH10:  JSR     PC,@#MODDSUB
17049 076046 047777 177777          1$:     .WORD   47777,-1        ;AC
17050 076052 177000 000000                  .WORD   177000,0
17051 076056 040200 000000 000000   2$:     .WORD   40200,0,0,0     ;FSRC
17052 076064 000000
17053 076066 000000 000000 000000   3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
17054 076074 000000
17055 076076 047777 177777          4$:     .WORD   47777,-1        ;INTEGER RES.
17056 076102 177000 000000                  .WORD   177000,0
17057 076106 000213                 7$:     213                     ;FPS BEFORE EXECUTION.
17058 076110 000204                         204                     ;FPS AFTER EXECUTION.
17059
17060                                       ;MODD TEST WITH EXPONENT OF THE RESULT = 0
17061 076112 004737 076306          HHH11:  JSR     PC,@#MODDSUB
17062 076116 040200 000000 000000   1$:     .WORD   40200,0,0,0     ;AC
17063 076124 000000
17064 076126 040177 072727          2$:     .WORD   40177,72727     ;FSRC
```

```
17065  076132  127272  072727                  .WORD   127272,72727
17066  076136  040177  072727          3$:     .WORD   40177,72727     ;FRACTIONAL RES.
17067  076142  127272  072727                  .WORD   127272,72727
17068  076146  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
17069  076154  000000
17070  076156  000200                  7$:     200                     ;FPS BEFORE EXECUTION.
17071  076160  000200                          200                     ;FPS AFTER EXECUTION.
17072
17073                                  ;MODD TEST WITH EXPONENT OF THE RESULT = -115
17074  076162  004737  076306          HHH12:  JSR     PC,@#MODDSUB
17075  076166  003377  177777          1$:     .WORD   3377,-1         ;AC
17076  076172  177777  052525                  .WORD   -1,52525
17077  076176  040200  000000  000000  2$:     .WORD   40200,0,0,0     ;FSRC
17078  076204  000000
17079  076206  003377  177777          3$:     .WORD   3377,-1         ;FRACTIONAL RES.
17080  076212  177777  052525                  .WORD   -1,52525
17081  076216  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
17082  076224  000000
17083  076226  000200                  7$:     200                     ;FPS BEFORE EXECUTION.
17084  076230  000200                          200                     ;FPS AFTER EXECUTION.
17085
17086                                  ;MODD TEST WITH EXPONENT OF THE RESULT = -63, IN ROUND MODE.
17087  076232  004737  076306          HHH13:  JSR     PC,@#MODDSUB
17088  076236  040300  000000  000000  1$:     .WORD   40300,0,0,0     ;AC
17089  076244  000000
17090  076246  020200  000000  000000  2$:     .WORD   20200,0,0,1     ;FSRC
17091  076254  000001
17092  076256  020300  000000  000000  3$:     .WORD   20300,0,0,2     ;FRACTIONAL RES.
17093  076264  000002
17094  076266  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
17095  076274  000000
17096  076276  000200                  7$:     200                     ;FPS BEFORE EXECUTION.
17097  076300  000200                          200                     ;FPS AFTER EXECUTION.
17098  076302  000137  076502          9$:     JMP     @#HHHDONE       ;GO TO THE NEXT TEST.
17099
17100                                  ;THIS SUBROUTINE, MODDSUB, IS CALLED TO SETUP THE
17101                                  ;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
17102                                  ;IT IS CALLED THUS:
17103                                  ;
17104                                  ;       ACARG:  .WORD   X,X,X,X         ;AC OPERAND
17105                                  ;       FSRCARG:.WORD   X,X,X,X         ;FSRC OPERAND
17106                                  ;       FRES:   .WORD   X,X,X,X         ;FRACTIONAL RESULT
17107                                  ;       INTRES: .WORD   X,X,X,X         ;INTEGER RESULT
17108                                  ;       ERFRES: .WORD   X,X,X,X         ;ERROR FRACTION RESULT
17109                                  ;       ERINTRES:.WORD  X,X,X,X         ;ERROR INTEGER RESULT
17110                                  ;       FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
17111                                  ;       FPSA:   .WORD   X               ;FPS AFTER EXECUTION
17112                                  ;       ERR1:   ERROR   X               ;FRACTION ERROR
17113                                  ;               BR      CONT
17114                                  ;       ERR2:   ERROR   X               ;INTEGER ERROR
17115                                  ;       CONT:                           ;RETURN ADDRESS
17116                                  ;
17117                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODD
17118                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
17119                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
17120                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
```

```
17121
17122
17123
17124
17125
17126
17127
17128
17129
17130
17131
17132
17133  076306  012601                 MODDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
17134  076310  012700  000200                 MOV     #200,R0         ;SET FD MODE.
17135  076314  170100                 LDFPS   R0
17136  076316  010100                 MOV     R1,R0           ;SET UP AC0
17137  076320  172410                 LDD     (R0),AC0
17138  076322  012700  075256         MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
17139  076326  172510                 LDD     (R0),AC1
17140  076330  016100  000040         MOV     40(R1),R0       ;SET UP THE FPS.
17141  076334  170100                 LDFPS   R0
17142  076336  010100                 MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
17143  076340  062700  000010         ADD     #10,R0
17144
17145  076344  171410                 1$:     MODD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
17146
17147  076346  170204                 STFPS   R4              ;GET THE FPS.
17148  076350  012700  000200         MOV     #200,R0         ;SET FD MODE.
17149  076354  170100                 LDFPS   R0
17150  076356  012700  076462         MOV     #MODDT0,R0      ;GET THE FRACTIONAL RESULT.
17151  076362  174010                 STD     AC0,(R0)
17152  076364  012700  076472         MOV     #MODDT1,R0      ;GET THE INTEGER RESULT.
17153  076370  174110                 STD     AC1,(R0)
17154  076372  012702  076462         MOV     #MODDT0,R2      ;CHECK THE FRACTIONAL RESULT.
17155  076376  010103                 MOV     R1,R3
17156  076400  062703  000020         ADD     #20,R3
17157  076404  012705  000004         MOV     #4,R5
17158  076410  022223                 2$:     CMP     (R2)+,(R3)+
17159  076412  001401                 BEQ     4$
17160  076414  104000                 EMT             ;
17161  076416  077504                 4$:     SOB     R5,2$
17162  076420  012702  076472         MOV     #MODDT1,R2      ;CHECK THE INTEGER RESULT.
17163  076424  010103                 MOV     R1,R3
17164  076426  062703  000030         ADD     #30,R3
17165  076432  012705  000004         MOV     #4,R5
17166  076436  022223                 3$:     CMP     (R2)+,(R3)+
17167  076440  001401                 BEQ     5$
17168  076442  104000                 EMT             ;
17169  076444  077504                 5$:     SOB     R5,3$
17170  076446  026104  000042         CMP     42(R1),R4       ;CHECK THE FPS.
17171  076452  001401                 BEQ     9$
17172  076454  104000                 EMT             ;
17173  076456  000161  000044         9$:     JMP     44(R1)          ;RETURN.
17174
17175  076462  000000  000000  000000  MODDT0: .WORD   0,0,0,0
17176  076470  000000
```

```
17177
17178  076472  000000  000000  000000  MODDT1: .WORD   0,0,0,0
17179  076500  000000
17180
17181  076502                          HHHDONE:
17182  076502  004767  021472                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
17183                                                                  ;SEE IF THE USER HAS EXPRESSED
17184                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
17185                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17186                                                                  ;THE USER TYPED CONTROL G?).
17187
17188
17189
17190
17191                                   ;*************************************************************************************
17192                                   ;TEST 475       UNDER\OVER FLOW, USING MODF WITH TRAPS DISABLED, TEST
17193                                   ;*************************************************************************************
17194  076506                           TS475:
17195
17196                                   ;UNDERFLOW TEST, WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1
17197  076506  004767  000214           MMM1:   JSR     PC,MODFOV
17198  076512  020123  045676           1$:     .WORD   20123,45676     ;AC
17199  076516  020200  000000           2$:     .WORD   20200,0         ;FSRC
17200  076522  000123  045676           3$:     .WORD   123,45676       ;FRACTIONAL RES.
17201  076526  000000  000000           4$:     .WORD   0,0             ;INTEGER RES.
17202  076532  042000                   7$:     42000                   ;FPS BEFORE EXECUTION.
17203  076534  142004                           142004                  ;FPS AFTER EXECUTION.
17204  076536  000012                           12                      ;FEC
17205  076540  104000                           EMT                     ;
17206
17207                                   ;UNDERFLOW EXP OF RESULT = -193, FIU = 0, FID = 1
17208  076542  004737  076726           MMM2:   JSR     PC,@#MODFOV
17209  076546  010200  000000           1$:     .WORD   10200,0         ;AC
17210  076552  010000  000000           2$:     .WORD   10000,0         ;FSRC
17211  076556  000000  000000           3$:     .WORD   0,0             ;FRACTIONAL RES.
17212  076562  000000  000000           4$:     .WORD   0,0             ;INTEGER RES.
17213  076566  005013                   7$:     5013                    ;FPS BEFORE EXECUTION.
17214  076570  005004                           5004                    ;FPS AFTER EXECUTION.
17215  076572  000012                           12                      ;FEC
17216  076574  000240                           NOP
17217
17218                                   ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
17219  076576  004737  076726           MMM3:   JSR     PC,@#MODFOV
17220  076602  060052  125252           1$:     .WORD   60052,125252    ;AC
17221  076606  060200  000000           2$:     .WORD   60200,0         ;FSRC
17222  076612  000000  000000           3$:     .WORD   0,0             ;FRACTIONAL RES.
17223  076616  000052  125252           4$:     .WORD   52,125252       ;INTEGER RES.
17224  076622  041000                   7$:     41000                   ;FPS BEFORE EXECUTION.
17225  076624  141006                           141006                  ;FPS AFTER EXECUTION.
17226  076626  000010                           10                      ;FEC
17227  076630                           8$:
17228  076630  104000                           EMT                     ;
17229                                   ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
17230  076632  004737  076726           MMM4:   JSR     PC,@#MODFOV
17231  076636  060345  067654           1$:     .WORD   60345,67654     ;AC
17232  076642  060200  000000           2$:     .WORD   60200,0         ;FSRC
```

```
17233  076646  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
17234  076652  000000  000000          4$:     .WORD   0,0             ;INTEGER RES.
17235  076656  006011                  7$:     6011                    ;FPS BEFORE EXECUTION.
17236  076660  006006                          6006                    ;FPS AFTER EXECUTION.
17237  076662  000010                          10                      ;FEC
17238  076664  000240                  8$:     NOP
17239                                  ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, RESULT NEGATIVE
17240                                  ;AND FIV = 1, FID = 1
17241  076666  004737  076726          MMM5:   JSR     PC,@#MODFOV
17242  076672  160252  125252          1$:     .WORD   160252,125252   ;AC
17243  076676  060000  000000          2$:     .WORD   60000,0         ;FSRC
17244  076702  000000  000000          3$:     .WORD   0,0             ;FRACTIONAL RES.
17245  076706  100052  125252          4$:     .WORD   100052,125252   ;INTEGER RES.
17246  076712  041000                  7$:     41000                   ;FPS BEFORE EXECUTION.
17247  076714  141006                          141006                  ;FPS AFTER EXECUTION.
17248  076716  000010                          10                      ;FEC
17249  076720                          8$:
17250  076720  104000                          EMT                     ;
17251  076722  000137  077142          9$:     JMP     @#MMMDONE       ;GO TO THE NEXT TEST.
17252
17253                                  ;THIS SUBROUTINE, MODFOV, IS CALLED TO SETUP THE
17254                                  ;OPERANDS, EXECUTE THE MODF INSTRUCTION AND CHECK THE RESULTS.
17255                                  ;IT IS CALLED THUS:
17256                                  ;
17257                                  ;               ACARG:  .WORD   X,X             ;AC OPERAND
17258                                  ;               FSRCARG:.WORD   X,X             ;FSRC OPERAND
17259                                  ;               FRES:   .WORD   X,X             ;FRACTIONAL RESULT
17260                                  ;               INTRES: .WORD   X,X             ;INTEGER RESULT
17261                                  ;               ERFRES: .WORD   X,X             ;ERROR FRACTION RESULT
17262                                  ;               ERINTRES:.WORD  X,X             ;ERROR INTEGER RESULT
17263                                  ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
17264                                  ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
17265                                  ;               FEC:    .WORD   X               ;FEC
17266                                  ;               ERR1:   ERROR   X               ;FEC ERROR
17267                                  ;                       BR      CONT
17268                                  ;               ERR2:   ERROR   X               ;INTEGER ERROR
17269                                  ;               CONT:                           ;RETURN ADDRESS
17270                                  ;
17271                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODF
17272                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
17273                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
17274                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
17275                                  ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
17276                                  ;THEN MODFOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
17277                                  ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
17278                                  ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
17279                                  ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
17280                                  ;FAILURE MATCHES THE TRUE RESULT THEN  MODFOV PASSES CONTROL TO THE
17281                                  ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
17282                                  ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
17283                                  ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
17284                                  ;IF A MATCH IS MADE HOWEVER, MODFOV WILL RETURN CONTROL TO THE ERROR
17285                                  ;CALL AT ERR2.
17286
17287  076726  012601                  MODFOV: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
17288  076730  012700  000200                  MOV     #200,R0         ;SET FD MODE.
```

```
17289  076734  170100                          LDFPS   R0
17290  076736  010100                          MOV     R1,R0           ;SET UP AC0
17291  076740  172410                          LDD     (R0),AC0
17292  076742  012700  075256                  MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
17293  076746  172510                          LDD     (R0),AC1
17294  076750  016100  000020                  MOV     20(R1),R0       ;SET UP THE FPS.
17295  076754  170100                          LDFPS   R0
17296  076756  010100                          MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
17297  076760  062700  000004                  ADD     #4,R0
17298
17299  076764  171410                  1$:     MODF    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
17300
17301  076766  170204                          STFPS   R4              ;GET THE FPS.
17302  076770  170305                          STST    R5              ;GET FEC.
17303  076772  012700  000200                  MOV     #200,R0         ;SET FD MODE.
17304  076776  170100                          LDFPS   R0
17305  077000  012700  077122                  MOV     #MODFD0,R0      ;GET THE FRACTIONAL RESULT.
17306  077004  174010                          STD     AC0,(R0)
17307  077006  012700  077132                  MOV     #MODFD1,R0      ;GET THE INTEGER RESULT.
17308  077012  174110                          STD     AC1,(R0)
17309  077014  012702  077122                  MOV     #MODFD0,R2      ;CHECK THE FRACTIONAL RESULT.
17310  077020  026112  000010                  CMP     10(R1),(R2)
17311  077024  001401                          BEQ     2$
17312  077026  104000                          EMT                     ;
17313  077030  026162  000012  000002  2$:     CMP     12(R1),2(R2)
17314  077036  001401                          BEQ     3$
17315  077040  104000                          EMT                     ;
17316  077042  012702  077132          3$:     MOV     #MODFD1,R2      ;CHECK THE INTEGER RESULT.
17317  077046  026112  000014                  CMP     14(R1),(R2)
17318  077052  001401                          BEQ     4$
17319  077054  104000                          EMT                     ;
17320  077056  026162  000016  000002  4$:     CMP     16(R1),2(R2)
17321  077064  001401                          BEQ     5$
17322  077066  104000                          EMT                     ;
17323  077070  026104  000022          5$:     CMP     22(R1),R4 '     ;CHECK THE FPS.
17324  077074  001401                          BEQ     6$
17325  077076  104000                          EMT                     ;
17326  077100  026105  000024          6$:     CMP     24(R1),R5       ;CHECK THE FEC.
17327  077104  001002                          BNE     25$             ;BRANCH IF INCORRECT.
17328
17329  077106  000161  000030          9$:     JMP     30(R1)          ;RETURN.
17330                                  ;REPORT FEC ERROR.
17331  077112  010102                  25$:    MOV     R1,R2
17332  077114  062702  000026                  ADD     #26,R2
17333  077120  000112                          JMP     (R2)
17334
17335  077122  000000  000000  000000  MODFD0: .WORD   0,0,0,0
17336  077130  000000
17337
17338  077132  000000  000000  000000  MODFD1: .WORD   0,0,0,0
17339  077140  000000
17340
17341  077142                          MMMDONE:
17342  077142  004767  021032                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
17343                                                                  ;SEE IF THE USER HAS EXPRESSED
17344                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
```

```
17345                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17346                                                   ;THE USER TYPED CONTROL G?).
17347
17348
17349
17350
17351                                   ;*****************************************************************************************
17352                                   ;TEST 476       UNDER\OVER FLOW, USING MODD WITH TRAPS DISABLED, TEST
17353                                   ;*****************************************************************************************
17354  077146                           TS476:
17355
17356                                   ;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -129, FIU = 1, FID = 1
17357  077146  004737  077432           NNN1:   JSR     PC,@#MODDOV
17358  077152  020252  125252           1$:     .WORD   20252,125252    ;AC
17359  077156  125252  125252                   .WORD   125252,125252
17360  077162  020100  000000  000000  2$:     .WORD   20100,0,0,0     ;FSRC
17361  077170  000000
17362  077172  000177  177777  177777  3$:     .WORD   177,-1,-1,-1    ;FRACTIONAL RES.
17363  077200  177777
17364  077202  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
17365  077210  000000
17366  077212  042200                   7$:     42200                   ;FPS BEFORE EXECUTION.
17367  077214  142204                           142204                  ;FPS AFTER EXECUTION.
17368  077216  000012                           12                      ;FEC
17369  077220                           8$:
17370  077220  104000                           EMT                     ;
17371                                   ;UNDERFLOW TEST WITH EXPONENT OF THE RESULT = -193, FIU = 0, FID = 1
17372  077222  004737  077432           NNN2:   JSR     PC,@#MODDOV
17373  077226  010000  000000           1$:     .WORD   10000,0         ;AC
17374  077232  123456  000000                   .WORD   123456,0
17375  077236  010200  000000  000000  2$:     .WORD   10200,0,0,0     ;FSRC
17376  077244  000000
17377  077246  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
17378  077254  000000
17379  077256  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
17380  077264  000000
17381  077266  005213                   7$:     5213                    ;FPS BEFORE EXECUTION.
17382  077270  005204                           5204                    ;FPS AFTER EXECUTION.
17383  077272  000012                           12
17384  077274  000240                   8$:     NOP
17385                                   ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 128, FIV = 1, FID = 1
17386  077276  004737  077432           NNN3:   JSR     PC,@#MODDOV
17387  077302  060252  125252           1$:     .WORD   60252,125252    ;AC
17388  077306  125252  125252                   .WORD   125252,125252
17389  077312  060100  000000  000000  2$:     .WORD   60100,0,0,0     ;FSRC
17390  077320  000000
17391  077322  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
17392  077330  000000
17393  077332  000177  177777  177777  4$:     .WORD   177,-1,-1,-1    ;INTEGER RES.
17394  077340  177777
17395  077342  041200                   7$:     41200                   ;FPS BEFORE EXECUTION.
17396  077344  141206                           141206                  ;FPS AFTER EXECUTION.
17397  077346  000010                           10                      ;FEC
17398  077350                           8$:
17399  077350  104000                           EMT                     ;
17400                                   ;OVERFLOW TEST WITH EXPONENT OF THE RESULT = 130, FIV = 0, FID = 1
```

```
17401  077352  004737  077432          NNN4:   JSR     PC,@#MODDOV
17402  077356  060200  000000          1$:     .WORD   60200,0         ;AC
17403  077362  125252  000000                  .WORD   125252,0
17404  077366  060200  000000  000000  2$:     .WORD   60200,0,0,0     ;FSRC
17405  077374  000000
17406  077376  000000  000000  000000  3$:     .WORD   0,0,0,0         ;FRACTIONAL RES.
17407  077404  000000
17408  077406  000000  000000  000000  4$:     .WORD   0,0,0,0         ;INTEGER RES.
17409  077414  000000
17410  077416  006211                  7$:     6211                    ;FPS BEFORE EXECUTION.
17411  077420  006206                          6206                    ;FPS AFTER EXECUTION.
17412  077422  000010                          10                      ;FEC
17413  077424  000240                  8$:     NOP
17414  077426  000137  077646          9$:     JMP     @#NNNDONE       ;GO TO NEXT TEST.
17415
17416                                  ;THIS SUBROUTINE, MODDOV, IS CALLED TO SETUP THE
17417                                  ;OPERANDS, EXECUTE THE MODD INSTRUCTION AND CHECK THE RESULTS.
17418                                  ;IT IS CALLED THUS:
17419                                  ;
17420                                  ;       ACARG:   .WORD   X,X,X,X        ;AC OPERAND
17421                                  ;       FSRCARG: .WORD   X,X,X,X        ;FSRC OPERAND
17422                                  ;       FRES:    .WORD   X,X,X,X        ;FRACTIONAL RESULT
17423                                  ;       INTRES:  .WORD   X,X,X,X        ;INTEGER RESULT
17424                                  ;       ERFRES:  .WORD   X,X,X,X        ;ERROR FRACTION RESULT
17425                                  ;       ERINTRES:.WORD   X,X,X,X        ;ERROR INTEGER RESULT
17426                                  ;       FPSB:    .WORD   X              ;FPS BEFORE EXECUTION
17427                                  ;       FPSA:    .WORD   X              ;FPS AFTER EXECUTION
17428                                  ;       ERR1:    ERROR   X              ;FRACTION ERROR
17429                                  ;                BR      CONT
17430                                  ;       ERR2:    ERROR   X              ;INTEGER ERROR
17431                                  ;       CONT:                           ;RETURN ADDRESS
17432                                  ;
17433                                  ;THE OPERANDS ARE SET UP (USING AC0 FOR THE AC ARGUMENT). THE MODD
17434                                  ;INSTRUCTION IS EXECUTED. THEN THE RESULTS ARE RETRIEVED.
17435                                  ;THE FRACTION PART OF THE RESULT IS COMPARED WITH FRES. IF THIS IS CORRECT
17436                                  ;THEN THE INTEGER PART IS COMPARED WITH INTRES. IF BOTH OF THESE ARE CORRECT
17437                                  ;THEN THE FPS IS COMPARED WITH FPSA. AFTER EXECUTION IF NO ERRORS OCCURRED
17438                                  ;THEN MODDOV WILL RETURN TO CONT. IF THE FPS WAS INCORRECT
17439                                  ;IT IS REPORTED HERE. IF THE FRACTION IS INCORRECT IT IS COMPARED WITH
17440                                  ;THE ANTICIPATED BAD FRACTION, ERFRES. IF THIS DOESN'T MATCH
17441                                  ;THE TRUE RESULT THEN THE ERROR IS REPORTED HERE. IF THE ANTICIPATED
17442                                  ;FAILURE MATCHES THE TRUE RESULT THEN  MODDOV PASSES CONTROL TO THE
17443                                  ;ERROR CALL AT ERR1. LIKEWISE IF THE INTEGER PART OF THE RESULT IS
17444                                  ;NOT CORRECT THEN IT IS COMPARED WITH THE ANTICIPATED INTEGER
17445                                  ;FAILURE. IF THIS DOEN'T MATCH THEN THE ERROR IS REPORTED HERE.
17446                                  ;IF A MATCH IS MADE HOWEVER, MODDOV WILL RETURN CONTROL TO THE ERROR
17447                                  ;CALL AT ERR2.
17448
17449  077432  012601                  MODDOV: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS
17450  077434  012700  000200                  MOV     #200,R0         ;SET FD MODE.
17451  077440  170100                          LDFPS   R0
17452  077442  010100                          MOV     R1,R0           ;SET UP AC0
17453  077444  172410                          LDD     (R0),AC0
17454  077446  012700  075256                  MOV     #MODP1,R0       ;PUT A BACKROUND PATTERN INTO AC1.
17455  077452  172510                          LDD     (R0),AC1
17456  077454  016100  000040                  MOV     40(R1),R0       ;SET UP THE FPS.
```

```
17457  077460  170100                    LDFPS   R0
17458  077462  010100                    MOV     R1,R0           ;COMPUTE THE ADDRESS OF THE FSRC.
17459  077464  062700  000010            ADD     #10,R0
17460
17461  077470  171410            1$:     MODD    (R0),AC0        ;EXECUTE THE TEST INSTRUCTION.
17462
17463  077472  170305                    STST    R5              ;GET THE FPS.
17464  077474  170204                    STFPS   R4              ;GET THE FPS.
17465  077476  012700  000200            MOV     #200,R0         ;SET FD MODE.
17466  077502  170100                    LDFPS   R0
17467  077504  012700  077626            MOV     #MODDD0,R0      ;GET THE FRACTIONAL RESULT.
17468  077510  174010                    STD     AC0,(R0)
17469  077512  012700  077636            MOV     #MODDD1,R0      ;GET THE INTEGER RESULT.
17470  077516  174110                    STD     AC1,(R0)
17471  077520  012702  077626            MOV     #MODDD0,R2      ;CHECK THE FRACTIONAL RESULT.
17472  077524  010103                    MOV     R1,R3
17473  077526  062703  000020            ADD     #20,R3
17474  077532  012700  000004            MOV     #4,R0
17475  077536  022223            2$:     CMP     (R2)+,(R3)+
17476  077540  001401                    BEQ     4$
17477  077542  104000                    EMT                     .
17478  077544  077004            4$:     SOB     R0,2$
17479  077546  012702  077636            MOV     #MODDD1,R2      ;CHECK THE INTEGER RESULT
17480  077552  010103                    MOV     R1,R3
17481  077554  062703  000030            ADD     #30,R3
17482  077560  012700  000004            MOV     #4,R0
17483  077564  022223            3$:     CMP     (R2)+,(R3)+
17484  077566  001401                    BEQ     5$
17485  077570  104000                    EMT                     ;
17486  077572  077004            5$:     SOB     R0,3$
17487  077574  026104  000042            CMP     42(R1),R4       ;CHECK THE FPS.
17488  077600  001401                    BEQ     6$
17489  077602  104000                    EMT                     ;
17490  077604  026105  000044    6$:     CMP     44(R1),R5       ;CHECK THE FEC.
17491  077610  001002                    BNE     25$
17492
17493  077612  000161  000050    9$:     JMP     50(R1)          ;RETURN.
17494                            ;REPORT FEC ERROR.
17495  077616  010102            25$:    MOV     R1,R2
17496  077620  062702  000046            ADD     #46,R2
17497  077624  000112                    JMP     (R2)
17498
17499  077626  000000  000000  000000  MODDD0: .WORD   0,0,0,0
17500  077634  000000
17501
17502  077636  000000  000000  000000  MODDD1: .WORD   0,0,0,0
17503  077644  000000
17504
17505  077646                    NNNDONE:
17506  077646  004767  020326            JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
17507                                                            ;SEE IF THE USER HAS EXPRESSED
17508                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
17509                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17510                                                            ;THE USER TYPED CONTROL G?).
17511
17512
```

```
                                ;*****************************************************************************
                                ;TEST 477       MORE MICROCODES COVERAGE
                                ;*****************************************************************************
077652                          TS477:
077652  012737  077666  000244  XT1:    MOV     #XT1A,@#244
077660  170227  000000          STFPS   #0
077664  000401                  BR      XT2
077666                          XT1A:
077666  104000                  EMT                             ;

                                XT2:    MOV     #-1,R0
077670  012700  177777
077674  170127  000000          LDFPS   #0
077700  170200                  STFPS   R0
077702  005700                  TST     R0
077704  001401                  BEQ     XT2A
077706  104000                  EMT                             ;
077710  012700  100444          XT2A:   MOV     #XPAT0,R0
077714  172440                  LDF     -(R0),AC0
077716  022700  100440          CMP     #XPAT0-4,R0
077722  001401                  BEQ     XT2B
077724  104000                  EMT                             ;
077726  170200                  XT2B:   STFPS   R0
077730  022700  000004          CMP     #4,R0                   ;CHECK IF FZ IS SET?
077734  001401                  BEQ     XT3
077736  104000                  EMT                             ;
077740  170127  000000          XT3:    LDFPS   #0
077744  012700  100444          MOV     #XPAT0,R0
077750  174040                  STF     AC0,-(R0)
077752  022700  100440          CMP     #XPAT0-4,R0
077756  001401                  BEQ     XT3A
077760  104000                  EMT                             ;
077762  170200                  XT3A:   STFPS   R0
077764  005700                  TST     R0
077766  001401                  BEQ     XT4
077770  104000                  EMT                             ;

077772  170127  000000          XT4:    LDFPS   #0
077776  012737  100022  000244  MOV     #XT4A,@#244
100004  170127  004000          LDFPS   #04000                  ;INTRPT ON UNDEFINED VARIABLE
100010  172437  100444          LDF     @#XPAT0,AC0
100014  174437  100474          DIVF    @#XPAT3,AC0             ;GET UNDEFINED VARIABLE, _0
100020  104000                  EMT                             ;
100022  170200                  XT4A:   STFPS   R0
100024  022700  104004          CMP     #104004,R0              ;CHECK: FER,FIUV,FZ ARE SET?
100030  001401                  BEQ     XT4B
100032  104000                  EMT                             ;
100034  012700  100434          XT4B:   MOV     #XBUF,R0
100040  174010                  STF     AC0,(R0)
100042  005737  100434          TST     @#XBUF
100046  001401                  BEQ     XT5
100050  104000                  EMT                             ;

100052  012737  100072  000244  XT5:    MOV     #XT5A,@#244
100060  170127  004000          LDFPS   #04000                  ;INTRPT ON UNDEFINED VARIBALE
100064  177437  100474          LDCDF   @#XPAT3,AC0             ;GET UNDEFINED VARIABLE, _0
100070  104000                  EMT                             ;
```

```
17569  100072  170200                  XT5A:   STFPS   R0
17570  100074  022700  104014                  CMP     #104014,R0      ;CHECK: FER,FIUV,FN,FZ ARE SET?
17571  100100  001401                          BEQ     XT5B
17572  100102  104000                          EMT                     ;
17573  100104  012700  100434          XT5B:   MOV     #XBUF,R0
17574  100110  174010                          STF     AC0,(R0)
17575  100112  005737  100434                  TST     @#XBUF
17576  100116  001401                          BEQ     XT6
17577  100120  104000                          EMT                     ;
17578
17579  100122  012737  100146  000244  XT6:    MOV     #XT6A,@#244
17580  100130  170127  004000                  LDFPS   #04000          ;INTRPT ON UNDEFINED VARIBALE
17581  100134  172437  100444                  LDF     @#XPAT0,AC0
17582  100140  172037  100474                  ADDF    @#XPAT3,AC0
17583  100144  104000                          EMT                     ;
17584  100146  170200                  XT6A:   STFPS   R0
17585  100150  022700  104004                  CMP     #104004,R0      ;CHECK: FER,FIUV,FZ ARE SET?
17586  100154  001401                          BEQ     XT6B
17587  100156  104000                          EMT                     ;
17588  100160  012700  100434          XT6B:   MOV     #XBUF,R0
17589  100164  174010                          STF     AC0,(R0)
17590  100166  005737  100434                  TST     @#XBUF
17591  100172  001401                          BEQ     XT7
17592  100174  104000                          EMT                     ;
17593
17594  100176  170127  000000          XT7:    LDFPS   #0
17595  100202  172437  100504                  LDF     @#XPAT4,AC0
17596  100206  175437  100534                  STCFI   AC0,@#XPAT0
17597  100212  022737  000002  100534          CMP     #2,@#XPAT0      ;CHECK DATA
17598  100220  001401                          BEQ     XT8
17599  100222  104000                          EMT                     ;
17600
17601  100224  170127  000100          XT8:    LDFPS   #100            ;SET FL
17602  100230  172437  100504                  LDF     @#XPAT4,AC0
17603  100234  175467  000274                  STCFI   AC0,XPAT0
17604  100240  022737  000002  100536          CMP     #2,@#XPAT0+2
17605  100246  001401                          BEQ     XT9
17606  100250  104000                          EMT                     ;
17607
17608                           ;START OF FPP2
17609  100252  170127  000000          XT9:    LDFPS   #0
17610  100256  172437  100444                  LDF     @#XPAT0,AC0
17611  100262  172037  100504                  ADDF    @#XPAT4,AC0
17612  100266  170200                          STFPS   R0
17613  100270  005700                          TST     R0
17614  100272  001401                          BEQ     XT10
17615  100274  104000                          EMT                     ;
17616
17617  100276  170127  000000          XT10:   LDFPS   #0
17618  100302  172437  100504                  LDF     @#XPAT4,AC0
17619  100306  173037  100504                  SUBF    @#XPAT4,AC0
17620  100312  170200                          STFPS   R0
17621  100314  022700  000004                  CMP     #4,R0
17622  100320  001401                          BEQ     XT11
17623  100322  104000                          EMT                     ;
17624
```

```
17625  100324  170127  000000          XT11:   LDFPS   #0
17626  100330  172437  100504                  LDF     @#XPAT4,AC0
17627  100334  173437  100504                  CMPF    @#XPAT4,AC0
17628  100340  170200                          STFPS   R0
17629  100342  022700  000004                  CMP     #4,R0           ;CHECK IF FZ IS SET?
17630  100346  001401                          BEQ     XT12
17631  100350  104000                          EMT                     ;
17632
17633  100352  170127  000000          XT12:   LDFPS   #0
17634  100356  172437  100504                  LDF     @#XPAT4,AC0
17635  100362  174437  100464                  DIVF    @#XPAT2,AC0
17636  100366  012700  100434                  MOV     #XBUF,R0
17637  100372  174010                          STF     AC0,(R0)
17638  100374  022737  040176  100434          CMP     #040176,@#XBUF  ;CHECK DATA
17639  100402  001401                          BEQ     XT13
17640  100404  104000                          EMT                     ;
17641
17642  100406  170127  000000          XT13:   LDFPS   #0
17643  100412  172437  100514                  LDF     @#XPAT5,AC0
17644  100416  174437  100524                  DIVF    @#XPAT6,AC0
17645  100422  170200                          STFPS   R0
17646  100424  022700  000004                  CMP     #4,R0
17647  100430  001445                          BEQ     XTDONE
17648  100432  104000                          EMT                     ;
17649
17650
17651  100434  000000  000000  000000  XBUF:   .WORD   0,0,0,0
17652  100442  000000
17653  100444  000000  000000  000000  XPAT0:  .WORD   0,0,0,0
17654  100452  000000
17655  100454  000001  000001  000001  XPAT1:  .WORD   1,1,1,1
17656  100462  000001
17657  100464  040401  000000  000000  XPAT2:  .WORD   40401,0,0,0
17658  100472  000000
17659  100474  100000  000000  000000  XPAT3:  .WORD   100000,0,0,0
17660  100502  000000
17661  100504  040400  000000  000000  XPAT4:  .WORD   040400,0,0,0
17662  100512  000000
17663  100514  000207  000000  000000  XPAT5:  .WORD   207,0,0,0
17664  100522  000000
17665  100524  077007  000000  000000  XPAT6:  .WORD   77007,0,0,0
17666  100532  000000
17667  100534  000000  000000  000000  XPAT0:  .WORD   0,0,0,0
17668  100542  000000
17669
17670  100544                          XTDONE:
17671  100544  004767  017430                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
17672                                                                  ;SEE IF THE USER HAS EXPRESSED
17673                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
17674                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17675                                                                  ;THE USER TYPED CONTROL G?).
17676
17677
17678
17679
17680                           ;*******************************************************************************
```

```
17681                                   ;TEST 500       STF WITH ILLEGAL ACCUMULATOR TEST
17682                                   ;*****************************************************************************
17683  100550                           TS500:
17684
17685  100550  005000                           CLR     R0              ;SET THE FPS.
17686  100552  170100                           LDFPS   R0
17687
17688  100554  012737  100574  000244           MOV     #0001,@#FPVECT  ;SET UP FOR FP TRAPS.
17689  100562  012737  100570  037244           MOV     #1$,@#$TMP2
17690
17691  100570  174007                   1$:     STF     AC0,AC7         ;THIS TEST INSTRUCTION SHOULD
17692                                                                   ;CAUSE A TRAP.
17693
17694                                   ;REPORT FAILURE OF USE OF ILLEGAL ACCUMULATOR 7 TO CAUSE AN FPP TRAP.
17695  100572                           0002:
17696  100572  104000                           EMT                     ;INSTRUCTION DID NOT TRAP
17697
17698                                   ;TRAP TO 0001, HERE, WHEN THE EXPECTED ERROR OCCURS.
17699  100574  011600                   0001:   MOV     (SP),R0         ;MAKE SURE THE ERROR OCCURRED
17700  100576  022700  100572                   CMP     #0002,R0        ;AT THE CORRECT ADDRESS.
17701  100602  001420                           BEQ     TS501
17702  100604  104000                           EMT                     ;FLOATING POINT TRAP DID NOT OPERATE RIGHT
17703
17704  100606  170204                   0003:   STFPS   R4              ;GET FPS.
17705  100610  170305                           STST    R5              ;GET FEC.
17706  100612  012702  100000                   MOV     #100000,R2      ;EXPECTED FPS
17707  100616  012703  000002                   MOV     #2,R3           ;EXPECTED FEC
17708  100622  022626                           CMP     (SP)+,(SP)+     ;RESET THE STACK.
17709
17710  100624  020204                           CMP     R2,R4           ;WAS FPS CORRECT?
17711  100626  001401                           BEQ     0004
17712  100630  104000                           EMT                     ;FPS INCORRECTLY SET AFTER USE OF ILLEGAL ACC
17713  100632  020305                   0004:   CMP     R3,R5           ;WAS THE FEC CORRECT?
17714  100634  001401                           BEQ     0001DONE
17715  100636  104000                           EMT                     ;INCORRECT FEC AFTER USE OF ILLEGAL ACC
17716
17717  100640                           0001DONE:
17718  100640  004767  017334                   JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
17719                                                                   ;SEE IF THE USER HAS EXPRESSED
17720                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
17721                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17722                                                                   ;THE USER TYPED CONTROL G?).
17723
17724
17725
17726
17727                                   ;*****************************************************************************
17728                                   ;TEST 501       FDST MODE 1, FLOATING MODE, TEST
17729                                   ;*****************************************************************************
17730  100644                           TS501:
17731
17732
17733  100644  012700  177777                   MOV     #-1,R0          ;SET UP A BACKROUND PATTERN IN THE
17734  100650  012701  100760                   MOV     #PPPBF0,R1      ;INPUT BUFFER.
17735  100654  012702  000014                   MOV     #14,R2
17736  100660  010021                   PPP2:   MOV     R0,(R1)+
```

```
17737  100662  077202                  SOB     R2,PPP2
17738
17739  100664  012700  000200          MOV     #200,R0         ;SET FD MODE.
17740  100670  170100                  LDFPS   R0
17741  100672  012700  101010          MOV     #PPPTP1,R0      ;PUT TEST DATA INTO AC0.
17742  100676  172410                  LDD     (R0),AC0
17743
```

```
17744  100700  012700  100774          MOV     #PPPBF1,R0      ;FDST ADDRESS.
17745  100704  005002                  CLR     R2              ;CLEAR THE FPS.
17746  100706  170102                  LDFPS   R2
17747
17748  100710  174010          PPP3:   STF     AC0,(R0)        ;TEST INSTRUCTION.
17749
17750  100712  022700  100774          CMP     #PPPBF1,R0      ;WAS R0 MODIFIED DURING EXECUTION?
17751  100716  001401                  BEQ     PPP4
17752  100720  104000                  EMT                     ;R0 MODIFIED
17753
17754  100722  012700  100774  PPP4:   MOV     #PPPBF1,R0      ;CHECK THE DATA IN THE OUTPUT BUFFER.
17755  100726  012701  101010          MOV     #PPPTP1,R1
17756  100732  022021                  CMP     (R0)+,(R1)+
17757  100734  001031                  BNE     PPP10           ;BRANCH IF INCORRECT.
17758  100736  022011                  CMP     (R0)+,(R1)
17759  100740  001027                  BNE     PPP10           ;BRANCH IF INCORRECT.
17760  100742  022720  177777          CMP     #-1,(R0)+       ;WAS FLOATING MODE USED?
17761  100746  001024                  BNE     PPP10           ;BRANCH IF NOT.
17762  100750  022710  177777          CMP     #-1,(R0)
17763  100754  001021                  BNE     PPP10
17764  100756  000421                  BR      PPPDONE ;GO TO NEXT TEST.
17765
17766  100760  177777  177777  177777  PPPBF0: .WORD   -1,-1,-1,-1,-1,-1
17767  100766  177777  177777  177777
17768
17769  100774  177777  177777  177777  PPPBF1: .WORD   -1,-1,-1,-1,-1,-1
17770  101002  177777  177777  177777
17771
17772  101010  123456  023456          PPPTP1: .WORD   123456,23456
17773  101014  034567  045671                  .WORD   34567,45671
17774
17775  101020                          PPP10:
17776  101020  104000                          EMT                     ;
17777  101022                          PPPDONE:
17778  101022  004767  017152                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
17779                                                                  ;SEE IF THE USER HAS EXPRESSED
17780                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
17781                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17782                                                                  ;THE USER TYPED CONTROL G?).
17783
17784
17785
17786
17787                           ;*************************************************************************************
17788                           ;TEST 502       FDST MODE 2 TEST
17789                           ;*************************************************************************************
17790  101026                  TS502:
17791
17792                           ;FIRST TEST STF.
17793
17794  101026  012700  177777          MOV     #-1,R0  ;SET UP THE OUTPUT BUFFER.
17795  101032  012701  101142          MOV     #QQQBF0,R1
17796  101036  012702  000014          MOV     #14,R2
17797  101042  010021          QQQ2:   MOV     R0,(R1)+
17798  101044  077202                  SOB     R2,QQQ2
17799
```

```
17800  101046  012700  000200          MOV     #200,R0         ;SET FD MODE.
17801  101052  170100                  LDFPS   R0
17802  101054  012700  101172          MOV     #QQQTP1,R0      ;SETUP AC0.
17803  101060  172410                  LDD     (R0),AC0
17804
17805  101062  012700  101156          MOV     #QQQBF1,R0      ;FDST ADDRESS.
17806  101066  005002                  CLR     R2
17807  101070  170102                  LDFPS   R2              ;SET FPS.
17808  101072  174020          QQQ3:   STF     AC0,(R0)+       ;TEST INSTRUCTION.
17809
17810  101074  022700  101162          CMP     #QQQBF1+4,R0    ;WAS R0 INCREMENTED BY 4 PROPERLY?
17811
17812  101100  001401                  BEQ     QQQ4
17813  101102  104000                  EMT                     ;REPORT R0 INCORRECT AFTER FDST MODE 2
17814  101104  012700  101156  QQQ4:   MOV     #QQQBF1,R0      ;WAS THE OUTPUT DATA CORRECT?
17815  101110  012701  101172          MOV     #QQQTP1,R1
17816  101114  022021                  CMP     (R0)+,(R1)+
17817  101116  001031                  BNE     QQQ10           ;BRANCH IF INCORRECT.
17818  101120  022021                  CMP     (R0)+,(R1)+
17819  101122  001027                  BNE     QQQ10           ;BRANCH IF INCORRECT.
17820  101124  022027  177777          CMP     (R0)+,#-1       ;SEE IF ANY OTHER DATA BUFFER WORDS WERE MODIFIED.
17821  101130  001024                  BNE     QQQ10           ;BRANCH IF INCORRECT.
17822  101132  022027  177777          CMP     (R0)+,#-1
17823  101136  001021                  BNE     QQQ10           ;BRANCH IF INCORRECT.
17824  101140  000421                  BR      QQQ20
17825  101142  177777  177777  177777 QQQBF0: .WORD   -1,-1,-1,-1,-1,-1
17826  101150  177777  177777  177777
17827  101156  177777  177777  177777 QQQBF1: .WORD   -1,-1,-1,-1,-1,-1
17828  101164  177777  177777  177777
17829  101172  076543          QQQTP1: 76543
17830  101174  065432                  65432
17831  101176  054321                  54321
17832  101200  043210                  43210
17833                          ;REPORT OUTPUT DATA INCORRECT:
17834  101202                  QQQ10:
17835  101202  104000                  EMT                     ;
17836
17837                          ;NOW TEST STD MODE 2.
17838
17839  101204  012700  101142  QQQ20:  MOV     #QQQBF0,R0      ;SET UP DEFAULT INPUT DATA BUFFER.
17840  101210  010001                  MOV     R0,R1
17841  101212  012702  000014          MOV     #14,R2
17842  101216  010021          QQQ22:  MOV     R0,(R1)+
17843  101220  077202                  SOB     R2,QQQ22
17844  101222  012700  000200          MOV     #200,R0         ;ENTER FLOATING DOUBLE MODE.
17845  101226  170100                  LDFPS   R0
17846  101230  012700  101172          MOV     #QQQTP1,R0      ;LOAD AC0.
17847  101234  172410                  LDD     (R0),AC0
17848  101236  012700  101156          MOV     #QQQBF1,R0      ;SET DESTINATION ADDRESS.
17849  101242  012737  101250  037244  MOV     #QQQ23,@#$TMP2
17850  101250  174020          QQQ23:  STD     AC0,(R0)+       ;TEST INSTRUCTION.
17851  101252  022700  101166          CMP     #QQQBF1+10,R0   ;WAS R0 INCREMENTED BY 10 CORRECTLY?
17852  101256  001401                  BEQ     QQQ24
17853  101260  104000                  EMT                     ;REPORT R0 INCORRECTLY INCREMENTED
17854  101262  012700  101156  QQQ24:  MOV     #QQQBF1,R0      ;DID THE DATA REACH THE OUTPUT BUFFER CORRECTLY?
17855  101266  012701  101172          MOV     #QQQTP1,R1
```

```
17856  101272  012702  000004          MOV     #4,R2
17857  101276  022021          1$:     CMP     (R0)+,(R1)+
17858  101300  001002                  BNE     QQQ25           ;BRANCH IF INCORRECT.
17859  101302  077203                  SOB     R2,1$
17860  101304  000401                  BR      QQQDONE
17861                          ;REPORT DATA INCORRECT.
17862  101306                  QQQ25:
17863  101306  104000                  EMT                     ;
17864  101310                  QQQDONE:
17865  101310  004767  016664          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
17866                                                          ;SEE IF THE USER HAS EXPRESSED
17867                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
17868                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17869                                                          ;THE USER TYPED CONTROL G?).
17870
17871                          ;*****************************************************************************
17872                          ;TEST 503       FDST MODE 2, WITH GR7, TEST
17873                          ;*****************************************************************************
17874  101314                  TS503:
17875
17876  101314  012700  101364          MOV     #RRR3,R0        ;SET UP THE DATA BUFFER FOLLOWING THE TEST INSTRUCTION.
17877  101320  012701  101432          MOV     #RRRTP1,R1
17878  101324  012702  000004          MOV     #4,R2
17879  101330  012021          1$:     MOV     (R0)+,(R1)+
17880  101332  077202                  SOB     R2,1$
17881  101334  012700  000200          MOV     #200,R0         ;ENTER FLOATING DOUBLE MODE.
17882  101340  170100                  LDFPS   R0
17883  101342  012700  101442          MOV     #RRRTP2,R0      ;SET UP AC0.
17884  101346  172410                  LDD     (R0),AC0
17885  101350  012737  101430  000004  MOV     #RRR10,@#ERRVECT ;SET UP FOR AN ODD ADDRESS.
17886  101356  005001                  CLR     R1
17887  101360  005004                  CLR     R4
17888                          ;THIS IS THE TEST INSTRUCTION. IT SHOULD MODIFY THE FIRST LOCATION
17889                          ;AFTER IT TO BE AN INCREMENT R4, INC R4, INSTRUCTION INSTEAD
17890                          ;OF AN INCREMENT R1 INSTRUCTION. THE INCREMENT R4 SHOULD NOT BE
17891                          ;EXECUTED SINCE THE PC SHOULD BE INCREMENTED BY TWO DURING IMMEDIATE
17892                          ;MODE ADDRESSING. THUS AFTER THE EXECUTION OF THE NEXT 5 INSTRUCTIONS
17893                          ;R1 SHOULD CONTAIN 3 AND R4 SHOULD CONTAIN 0.
17894  101362  174027          RRR2:   STD     AC0,(R7)+       ;TEST INSTRUCTION.
17895  101364  005201          RRR3:   INC     R1              ;THE STD INSTRUCTION SHOULD CHANGE THIS TO INC R4.
17896  101366  005201                  INC     R1
17897  101370  005201                  INC     R1
17898  101372  005201                  INC     R1
17899  101374  012700  101452          MOV     #RRREXP,R0      ;SEE IF THE DATA WAS OUTPUT CORRECTLY.
17900  101400  012702  101364          MOV     #RRR3,R2
17901  101404  012703  000004          MOV     #4,R3
17902  101410  022022          RRR4:   CMP     (R0)+,(R2)+
17903  101412  001006                  BNE     RRR10           ;BRANCH IF INCORRECT.
17904  101414  077303                  SOB     R3,RRR4
17905  101416  005704                  TST     R4              ;MAKE SURE R4 IS 0.
17906  101420  001003                  BNE     RRR10           ;BRANCH IF R4 IS INCORRECT.
17907  101422  022701  000003          CMP     #3,R1           ;SEE IF R1 IS CORRECT.
17908  101426  001415                  BEQ     RRRDONE
17909  101430                  RRR10:
17910  101430  104000                  EMT                     ;
17911                          ;THESE ARE TEST DATA PATTERNS USED TO SET UP THE OUTPUT BUFFER AT RRR3.
```

```
17912  101432  005201                  RRRTP1: INC     R1
17913  101434  005201                          INC     R1
17914  101436  005201                          INC     R1
17915  101440  005201                          INC     R1
17916                                  ;THIS IS THE DATA PUT IN AC0 BEFORE EXECUTION OF THE STD.
17917  101442  005204                  RRRTP2: INC     R4
17918  101444  005204                          INC     R4
17919  101446  005204                          INC     R4
17920  101450  005204                          INC     R4
17921                                  ;THIS IS THE EXPECTED DATA AT RRR3 AFTER EXECUTION OF THE STD.
17922  101452  005204                  RRREXP: INC     R4
17923  101454  005201                          INC     R1
17924  101456  005201                          INC     R1
17925  101460  005201                          INC     R1
17926  101462                          RRRDONE:
17927  101462  004767  016512                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
17928                                                                  ;SEE IF THE USER HAS EXPRESSED
17929                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
17930                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17931                                                                  ;THE USER TYPED CONTROL G?).
17932
17933  ;*****************************************************************************************
17934  ;TEST 504       FDST MODE 4 TEST
17935  ;*****************************************************************************************
17936  101466                          TS504:
17937
17938  101466  012700  177777                  MOV     #-1,R0          ;SET UP THE OUTPUT BUFFER.
17939  101472  012701  101614                  MOV     #SSSBF0,R1
17940  101476  012702  000010                  MOV     #10,R2
17941  101502  010021                  1$:     MOV     R0,(R1)+
17942  101504  077202                          SOB     R2,1$
17943  101506  012700  000200                  MOV     #200,R0         ;ENTER FLOATING DOUBLE MODE.
17944  101512  170100                          LDFPS   R0
17945  101514  012700  101634                  MOV     #SSSTP1,R0      ;SET UP AC0.
17946  101520  172410                          LDD     (R0),AC0
17947  101522  012737  101644  000004          MOV     #SSS10,@#ERRVECT ;SET UP FOR A TRAP TO 4.
17948  101530  012700  101624                  MOV     #SSSA1,R0       ;SET UP THE DESTINATION ADDRESS.
17949
17950  101534  174040                  SSS2:   STD     AC0,-(R0)       ;TEST INSTRUCTION.
17951  101536  005201                          INC     R1
17952  101540  020027  101614                  CMP     R0,#SSSBF0      ;SEE IF R0 WAS DECREMENTED PROPERLY.
17953  101544  001037                          BNE     SSS10           ;BRANCH IF R0 IS INCORRECT.
17954  101546  012700  101614                  MOV     #SSSBF0,R0      ;WAS THE OUTPUT DATA CORRECT?
17955  101552  012701  101634                  MOV     #SSSTP1,R1
17956  101556  012702  000004                  MOV     #4,R2
17957  101562  022021                  1$:     CMP     (R0)+,(R1)+
17958  101564  001027                          BNE     SSS10           ;BRANCH IF INCORRECT.
17959  101566  077203                          SOB     R2,1$
17960  101570  012700  177777                  MOV     #-1,R0          ;IS THE REST OF THE OUTPUT BUFFER CORRECT, -1?
17961  101574  012701  101624                  MOV     #SSSA1,R1
17962  101600  012702  000004                  MOV     #4,R2
17963  101604  020021                  2$:     CMP     R0,(R1)+
17964  101606  001016                          BNE     SSS10           ;BRANCH IF INCORRECT.
17965  101610  077203                          SOB     R2,2$
17966  101612  000415                          BR      SSSDONE
17967
```

```
17968
17969  101614  177777                ;THIS IS THE OUTPUT DATA BUFFER.
17970  101616  177777          SSSBF0: -1
17971  101620  177777                  -1
17972  101622  177777                  -1
17973  101624  177777                  -1
17974  101626  177777          SSSA1:  -1
17975  101630  177777                  -1
17976  101632  177777                  -1
17977
17978                          ;THIS IS THE TEST DATA LOADED INTO AC0:
17979  101634  147250          SSSTP1: 147250
17980  101636  036147                  36147
17981  101640  025036                  25036
17982  101642  147250                  147250
17983
17984  101644                  SSS10:
17985  101644  104000                  EMT                     ;
17986  101646                  SSSDONE:
17987  101646  004767  016326          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
17988                                                          ;SEE IF THE USER HAS EXPRESSED
17989                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
17990                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
17991                                                          ;THE USER TYPED CONTROL G?).
17992
17993          ;*****************************************************************************************
17994          ;TEST 505       FDST MODE 3 TEST
17995          ;*****************************************************************************************
17996  101652  TS505:
17997
17998  101652  012701  101762          MOV     #TTTBF0,R1      ;SET UP THE OUTPUT DATA BUFFER.
17999  101656  012700  177777          MOV     #-1,R0
18000  101662  012702  000013          MOV     #13,R2
18001  101666  010021          1$:     MOV     R0,(R1)+
18002  101670  077202                  SOB     R2,1$
18003  101672  012737  101762  101776  MOV     #TTTBF0,@#TTTA2
18004  101700  012700  000200          MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
18005  101704  170100                  LDFPS   R0
18006  101706  012700  102000          MOV     #TTTTP1,R0      ;SET UP AC0.
18007  101712  172410                  LDD     (R0),AC0
18008  101714  012737  102010  000004  MOV     #TTT10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
18009  101722  012700  101776          MOV     #TTTA2,R0       ;SET UP THE DESTINATION ADDRESS.
18010
18011  101726  174030          TTT2:   STD     AC0,@(R0)+      ;TEST INSTRUCTION.
18012
18013  101730  020027  102000          CMP     R0,#TTTA2+2     ;SEE IF R0 WAS INCREMENTED CORRECTLY.
18014  101734  001025                  BNE     TTT10           ;BRANCH IF INCORRECT.
18015  101736  012701  101762          MOV     #TTTBF0,R1      ;CHECK THE OUTPUT DATA BUFFER.
18016  101742  012702  102000          MOV     #TTTTP1,R2
18017  101746  012703  000004          MOV     #4,R3
18018  101752  022122          TTT3:   CMP     (R1)+,(R2)+
18019  101754  001015                  BNE     TTT10           ;BRANCH IF NOT CORRECT.
18020  101756  077303                  SOB     R3,TTT3
18021  101760  000414                  BR      TTTDONE
18022
18023                          ;THIS IS THHE OUTPUT DATA BUFFER:
```

```
18024  101762  177777                  TTTBF0: -1
18025  101764  177777                          -1
18026  101766  177777                          -1
18027  101770  177777                          -1
18028  101772  177777                          -1
18029  101774  177777                  TTTA1:  -1
18030  101776  101762                  TTTA2:  TTTBF0
18031  102000  101213                  TTTTP1: 101213
18032  102002  141516                          141516
18033  102004  071727                          71727
18034  102006  037475                          37475
18035
18036  102010                          TTT10:
18037  102010  104000                          EMT                     ;
18038
18039  102012                          TTTDONE:
18040  102012  004767  016162                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
18041                                                                  ;SEE IF THE USER HAS EXPRESSED
18042                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18043                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18044                                                                  ;THE USER TYPED CONTROL G?).
18045
18046                                  ;*****************************************************************************
18047                                  ;TEST 506       FDST MODE 5 TEST
18048                                  ;*****************************************************************************
18049  102016                          TS506:.
18050
18051  102016  012701  102126                  MOV     #UUUBF0,R1      ;SET UP THE OUTPUT DATA BUFFER.
18052  102022  012700  177777                  MOV     #-1,R0
18053  102026  012702  000013                  MOV     #13,R2
18054  102032  010021                  1$:     MOV     R0,(R1)+
18055  102034  077202                          SOB     R2,1$
18056  102036  012737  102126  102140          MOV     #UUUBF0,@#UUUA1
18057  102044  012700  000200                  MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
18058  102050  170100                          LDFPS   R0
18059  102052  012700  102144                  MOV     #UUUTP1,R0      ;SET UP AC0.
18060  102056  172410                          LDD     (R0),AC0
18061  102060  012737  102154  000004          MOV     #UUU10,@#ERRVECT ;GET READY FOR ANY TRAPS TO 4.
18062  102066  012700  102142                  MOV     #UUUA2,R0       ;SET UP THE DESTINATION ADDRESS.
18063  102072  174050                  UUU2:   STD     AC0,@-(R0)      ;TEST INSTRUCTION.
18064  102074  020027  102140                  CMP     R0,#UUUA2-2     ;WAS R0 DECRIMENTED PROPERLY?
18065  102100  001025                          BNE     UUU10           ;BRANCH IF R0 IS INCORRECT.
18066  102102  012701  102126                  MOV     #UUUBF0,R1      ;WAS THE DATA OUTPUT CORRECTLY?
18067  102106  012702  102144                  MOV     #UUUTP1,R2
18068  102112  012703  000004                  MOV     #4,R3
18069  102116  022122                  UUU3:   CMP     (R1)+,(R2)+
18070  102120  001015                          BNE     UUU10           ;BRANCH IF DATA IS INCORRECT.
18071  102122  077303                          SOB     R3,UUU3
18072  102124  000414                          BR      UUUDONE
18073
18074                                  ;THIS IS THE OUTPUT DATA BUFFER
18075  102126  177777                  UUUBF0: -1
18076  102130  177777                          -1
18077  102132  177777                          -1
18078  102134  177777                          -1
18079  102136  177777                          -1
```

```
18080  102140  102126                  UUUA1:  UUUBF0
18081  102142  177777                  UUUA2:  -1
18082  102144  020212                  UUUTP1: 20212
18083  102146  023242                          23242
18084  102150  026273                          26273
18085  102152  031323                          031323
18086
18087  102154                          UUU10:
18088  102154  104000                          EMT                     ;
18089  102156                          UUUDONE:
18090  102156  004767  016016                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18091                                                                  ;SEE IF THE USER HAS EXPRESSED
18092                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18093                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18094                                                                  ;THE USER TYPED CONTROL G?).
18095
18096                                  ;*****************************************************************************
18097                                  ;TEST 507       FDST MODE 6, INDEX MODE, TEST
18098                                  ;*****************************************************************************
18099  102162                          TS507:
18100
18101  102162  012700  000200                  MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
18102  102166  170100                          LDFPS   R0
18103  102170  012701  102272                  MOV     #VVVBF0,R1      ;SET UP THE OUT PUT DATA BUFFER.
18104  102174  012700  177777                  MOV     #-1,R0
18105  102200  012702  000004                  MOV     #4,R2
18106  102204  010021                  1$:     MOV     R0,(R1)+
18107  102206  077202                          SOB     R2,1$
18108  102210  012737  102312  000004        MOV     #VVV10,@#ERRVECT ;SET UP VECTOR 4 INCASE OF ERROR.
18109  102216  012700  102302                  MOV     #VVVTP1,R0      ;SET UP AC0.
18110  102222  172410                          LDD     (R0),AC0
18111  102224  012700  074371                  MOV     #VVVBF0-5701,R0 ;SET UP THE DESTINATION ADDRESS.
18112  102230  012701  000001                  MOV     #1,R1
18113  102234  174060  005701          VVV2:   STD     AC0,5701(R0)    ;TEST INSTRUCTION.
18114
18115  102240  020027  074371                  CMP     R0,#VVVBF0-5701 ;SEE IF R0 WAS MODIFIED.
18116  102244  001022                          BNE     VVV10           ;BRANCH IF INCORRECT.
18117  102246  012702  102272                  MOV     #VVVBF0,R2      ;WAS THE OUTPUT DATA CORRECT.
18118  102252  012703  102302                  MOV     #VVVTP1,R3
18119  102256  012704  000004                  MOV     #4,R4
18120  102262  022223                  1$:     CMP     (R2)+,(R3)+
18121  102264  001012                          BNE     VVV10           ;BRANCH IF INCORRECT DATA.
18122  102266  077403                          SOB     R4,1$
18123  102270  000411                          BR      VVVDONE
18124  102272  177777          VVVBF0: -1
18125  102274  177777                          -1
18126  102276  177777                          -1
18127  102300  177777                          -1
18128  102302  030313          VVVTP1: 30313
18129  102304  023334                          23334
18130  102306  035363                          35363
18131  102310  074041                          74041
18132
18133  102312                          VVV10:
18134  102312  104000                          EMT                     ;
18135  102314                          VVVDONE:
```

```
18136  102314  004767  015660                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
18137                                                                   ;SEE IF THE USER HAS EXPRESSED
18138                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
18139                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18140                                                                   ;THE USER TYPED CONTROL G?).
18141
18142                                   ;*****************************************************************************
18143                                   ;TEST 510       FDST MODE 7, INDEX DEFERRED MODE, TEST
18144                                   ;*****************************************************************************
18145  102320                           TS510:
18146
18147  102320  012700  000200                  MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
18148  102324  170100                          LDFPS   R0
18149  102326  012701  102436                  MOV     #WWWBF0,R1      ;SET UP THE OUTPUT DATA BUFFER.
18150  102332  012700  177777                  MOV     #-1,R0
18151  102336  012702  000004                  MOV     #4,R2
18152  102342  010021                  1$:     MOV     R0,(R1)+
18153  102344  077202                          SOB     R2,1$
18154  102346  012737  102466  000004         MOV     #WWW10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
18155  102354  012700  102446                  MOV     #WWWTP1,R0      ;SET UP AC0.
18156  102360  172410                          LDD     (R0),AC0
18157  102362  012700  074555                  MOV     #WWWBF1-5701,R0 ;SET UP THE DESTINATION ADDRESS.
18158  102366  012701  000001                  MOV     #1,R1
18159  102372  012737  102436  102456         MOV     #WWWBF0,@#WWWBF1
18160  102400  174070  005701          WWW2:   STD     AC0,@5701(R0)   ;TEST INSTRUCTION.
18161
18162  102404  020027  074555                  CMP     R0,#WWWBF1-5701 ;IS R0 CORRECT?
18163  102410  001026                          BNE     WWW10           ;BRANCH IF INCORRECT.
18164  102412  012702  102436                  MOV     #WWWBF0,R2      ;WAS THE DATA OUTPUT CORRECTLY?
18165  102416  012703  102446                  MOV     #WWWTP1,R3
18166  102422  012704  000004                  MOV     #4,R4
18167  102426  022223                  1$:     CMP     (R2)+,(R3)+
18168  102430  001016                          BNE     WWW10           ;BRANCH IF DATA IS INCORRECT.
18169  102432  077403                          SOB     R4,1$
18170  102434  000415                          BR      WWWDONE
18171  102436  177777                  WWWBF0: -1
18172  102440  177777                          -1
18173  102442  177777                          -1
18174  102444  177777                          -1
18175  102446  041424                  WWWTP1: 41424
18176  102450  034445                          34445
18177  102452  046475                          46475
18178  102454  051525                          051525
18179  102456  177777                  WWWBF1: -1
18180  102460  177777                          -1
18181  102462  177777                          -1
18182  102464  177777                          -1
18183
18184  102466                           WWW10:
18185  102466  104000                          EMT                     ;
18186  102470                           WWWDONE:
18187  102470  004767  015504                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
18188                                                                   ;SEE IF THE USER HAS EXPRESSED
18189                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
18190                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18191                                                                   ;THE USER TYPED CONTROL G?).
```

```
18192
18193                                   ;*****************************************************************************************
18194                                   ;TEST 511       STCFD TEST
18195                                   ;*****************************************************************************************
18196  102474                           TS511:
18197
18198                                   ;AC=0
18199  102474  004767  000262          XXX1:   JSR     PC,STCFDS
18200  102500  000000                  1$:     0               ;AC
18201  102502  000000                          0
18202  102504  000000                          0
18203  102506  000000                          0
18204  102510  000000                  2$:     0               ;RES
18205  102512  000000                          0
18206  102514  000000                          0
18207  102516  000000                          0
18208  102520  000000                  3$:     0               ;ERROR RES.
18209  102522  000000                          0
18210  102524  177777                          -1
18211  102526  177777                          -1
18212  102530  047000                  4$:     47000           ;FPS BEFORE EXECUTION.
18213  102532  047004                          47004           ;FPS AFTER EXECUTION.
18214  102534  177777                          -1              ;FEC
18215  102536  147004                          147004          ;ERROR FPS.
18216
18217
18218  102540  004767  000216          XXX2:   JSR     PC,STCFDS
18219  102544  017203                  1$:     17203           ;AC
18220  102546  142536                          142536
18221  102550  047506                          47506
18222  102552  172031                          172031
18223  102554  017203                  2$:     17203           ;RES
18224  102556  142536                          142536
18225  102560  000000                          0
18226  102562  000000                          0
18227  102564  017203                  3$:     17203           ;ERROR RES.
18228  102566  142536                          142536
18229  102570  047506                          47506
18230  102572  172031                          172031
18231  102574  040000                  4$:     40000           ;FPS BEFORE EXECUTION.
18232  102576  040000                          40000           ;FPS AFTER EXECUTION.
18233  102600  177777                          -1              ;FEC
18234  102602  177777                          -1              ;ERROR FPS.
18235
18236
18237  102604  004767  000152          XXX3:   JSR     PC,STCFDS
18238  102610  050717                  1$:     50717           ;AC
18239  102612  027374                          27374
18240  102614  075767                          75767
18241  102616  077071                          77071
18242  102620  050717                  2$:     50717           ;RES
18243  102622  027374                          27374
18244  102624  000000                          0
18245  102626  000000                          0
18246  102630  000000                  3$:     0               ;ERROR RES.
18247  102632  000000                          0
```

```
18248  102634  000000                          0
18249  102636  000000                          0
18250  102640  047000                  4$:     47000           ;FPS BEFORE EXECUTION.
18251  102642  047000                          47000           ;FPS AFTER EXECUTION.
18252  102644  177777                          -1              ;FEC
18253  102646  174002                          174002          ;ERROR FPS.
18254
18255
18256  102650  004767  000106          XXX4:   JSR     PC,STCFDS
18257  102654  020212                  1$:     20212           ;AC
18258  102656  032425                          32425
18259  102660  026272                          26272
18260  102662  002123                          02123
18261  102664  020212                  2$:     20212           ;RES
18262  102666  032425                          32425
18263  102670  000000                          0
18264  102672  000000                          0
18265  102674  020212                  3$:     20212           ;ERROR RES.
18266  102676  032425                          32425
18267  102700  100000                          100000
18268  102702  000000                          0
18269  102704  040000                  4$:     40000           ;FPS BEFORE EXECUTION.
18270  102706  040000                          40000           ;FPS AFTER EXECUTION.
18271  102710  177777                          -1              ;FEC
18272  102712  177777                          -1              ;ERROR FPS.
18273
18274
18275  102714  004767  000042          XXX5:   JSR     PC,STCFDS
18276  102720  121314                  1$:     121314          ;AC
18277  102722  151617                          151617
18278  102724  101112                          101112
18279  102726  131415                          131415
18280  102730  121314                  2$:     121314          ;RES
18281  102732  151617                          151617
18282  102734  000000                          0
18283  102736  000000                          0
18284  102740  021314                  3$:     21314           ;ERROR RES.
18285  102742  151617                          151617
18286  102744  000000                          0
18287  102746  000000                          0
18288  102750  040000                  4$:     40000           ;FPS BEFORE EXECUTION.
18289  102752  040010                          40010           ;FPS AFTER EXECUTION.
18290  102754  177777                          -1              ;FEC
18291  102756  177777                          -1              ;ERROR FPS.
18292  102760  000460                  6$:     BR      XXXDONE
18293
18294                                  ;
18295
18296
18297                                  ;THIS SUBROUTINE, STCFDS, IS USED TO SET UP THE OPERANDS, EXECUTE
18298                                  ;THE STCFD INSTRUCTION AND CHECK THE RESULTS. A CALL
18299                                  ;TO IT IS MADE THUS:
18300                                  ;
18301                                  ;               JSR     PC,@#STCFDS
18302                                  ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
18303                                  ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
```

```
18304
18305
18306
18307
18308
18309
18310
18311
18312
18313
18314
18315
18316
18317
18318
18319
18320
18321
18322
18323
18324
18325
18326
18327
18328
18329  102762  012601
18330  102764  012700  000200
18331  102770  170100
18332  102772  010100
18333  102774  172410
18334  102776  012700  177777
18335  103002  012702  103112
18336  103006  012703  000004
18337  103012  010022
18338  103014  077302
18339  103016  016100  000030
18340  103022  170100
18341  103024  012700  103112
18342  103030  176010
18343
18344  103032  170204
18345  103034  170305
18346  103036  010102
18347  103040  062702  000010
18348  103044  012703  103112
18349  103050  012700  000004
18350  103054  022223
18351  103056  001014
18352  103060  077003
18353
18354  103062  016102  000032
18355  103066  020204
18356  103070  001007
18357  103072  005702
18358  103074  100003
18359  103076  026105  000036
```

```
;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
;               FEC:    .WORD   X               ;EXPECTED FEC
;               ERFPS:  .WORD   X               ;ERROR FPS.
;               ERR1:   ERROR   X               ;DATA ERROR.
;                       BR      CONT
;               ERR2:   ERROR   X               ;FPS ERROR.
;               CONT:                           ;RETURN ADDRESS
;
;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
;THE STCFD INSTRUCTION IS EXECUTED.
;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCFDS RETURNS CONTROL
;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCFDS
;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCFDS WILL RETURN
;TO THE ERROR CALL AT ERR2, OTHERWISE STCFDS ITSELF
;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
;STCFD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCFDS
;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

STCFDS: MOV     (SP)+,R1                ;PICK UP THE POINTER TO THE OPERANDS.
        MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
        LDFPS   R0
        MOV     R1,R0           ;LOAD AC0.
        LDD     (R0),AC0
        MOV     #-1,R0          ;FILL THE OUTPUT BUFFER WITH -1'S.
        MOV     #STCFT,R2
        MOV     #4,R3
1$:     MOV     R0,(R2)+
        SOB     R3,1$
        MOV     30(R1),R0       ;LOAD THE FPS.
        LDFPS   R0
        MOV     #STCFT,R0       ;SET UP THE DESTINATION ADDRESS.
2$:     STCFD   AC0,(R0)        ;TEST INSTRUCTION.

        STFPS   R4              ;GET THE FPS.
        STST    R5              ;GET THE FEC.
        MOV     R1,R2           ;CHECK THE RESULT.
        ADD     #10,R2
        MOV     #STCFT,R3
        MOV     #4,R0
3$:     CMP     (R2)+,(R3)+
        BNE     10$             ;BRANCH IF INCORRECT.
        SOB     R0,3$

        MOV     32(R1),R2
        CMP     R2,R4           ;IS THE FPS CORRECT?
        BNE     10$             ;BRANCH IF FPS INCORRECT.
        TST     R2              ;IF EXPECTED FPS IS NEGATIVE, THEN
        BPL     4$              ;GO AHEAD AND CHECK THE FEC.
        CMP     36(R1),R5
```

```
18360  103102  001002                          BNE     10$             ;BRANCH IF FEC IS INCORRECT.
18361  103104  000161  000040          4$:     JMP     40(R1)          ;RETURN.
18362  103110                          10$:
18363  103110  104000                          EMT                     ;
18364  103112  177777  177777  177777  STCFT:  -1,-1,-1,-1
18365  103120  177777
18366  103122                          XXXDONE:
18367  103122  004767  015052                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18368                                                                  ;SEE IF THE USER HAS EXPRESSED
18369                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18370                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18371                                                                  ;THE USER TYPED CONTROL G?).
18372
18373                                  ;*****************************************************************************
18374                                  ;TEST 512       STCDF TEST
18375                                  ;*****************************************************************************
18376  103126                          TS512:
18377
18378                                  ;AC=0
18379  103126  004767  000262          YYY1:   JSR     PC,STCDFS
18380  103132  000000                  1$:     0                       ;AC
18381  103134  000000                          0
18382  103136  000000                          0
18383  103140  000000                          0
18384  103142  000000                  2$:     0                       ;RES
18385  103144  000000                          0
18386  103146  177777                          -1
18387  103150  177777                          -1
18388  103152  000000                  3$:     0                       ;ERROR RES.
18389  103154  000000                          0
18390  103156  000000                          0
18391  103160  000000                          0
18392  103162  047200                  4$:     47200                   ;FPS BEFORE EXECUTION.
18393  103164  047204                          47204                   ;FPS AFTER EXECUTION.
18394  103166  177777                          -1                      ;FEC
18395  103170  177777                          -1                      ;ERROR FPS.
18396
18397
18398  103172  004767  000216          YYY2:   JSR     PC,STCDFS
18399  103176  067574                  1$:     67574                   ;AC0
18400  103200  073727                          73727
18401  103202  170777                          170777
18402  103204  067574                          67574
18403  103206  067574                  2$:     67574                   ;RES
18404  103210  073730                          73730
18405  103212  177777                          -1
18406  103214  177777                          -1
18407  103216  067574                  3$:     67574                   ;ERROR RES.
18408  103220  073727                          73727
18409  103222  177777                          -1
18410  103224  177777                          -1
18411  103226  040200                  4$:     40200                   ;FPS BEFORE EXECUTION.
18412  103230  040200                          40200                   ;FPS AFTER EXECUTION.
18413  103232  177777                          -1                      ;FEC
18414  103234  177777                          -1                      ;ERROR FPS.
18415
```

```
18416
18417  103236  004767  000152          YYY3:   JSR     PC,STCDFS
18418  103242  077777                  1$:     77777                   ;AC0
18419  103244  177777                          -1
18420  103246  100000                          100000
18421  103250  000000                          0
18422  103252  000000                  2$:     0                       ;RES
18423  103254  000000                          0
18424  103256  177777                          -1
18425  103260  177777                          -1
18426  103262  077777                  3$:     77777                   ;ERROR RES.
18427  103264  177777                          -1
18428  103266  177777                          -1
18429  103270  177777                          -1
18430  103272  040200                  4$:     40200                   ;FPS BEFORE EXECUTION.
18431  103274  040206                          40206                   ;FPS AFTER EXECUTION.
18432  103276  177777                          -1                      ;FEC
18433  103300  040204                          40204                   ;ERROR FPS.
18434
18435
18436  103302  004767  000106          YYY4:   JSR     PC,STCDFS
18437  103306  077777                  1$:     77777                   ;AC0
18438  103310  177777                          -1
18439  103312  100000                          100000
18440  103314  000000                          0
18441  103316  000000                  2$:     0                       ;RES
18442  103320  000000                          0
18443  103322  177777                          -1
18444  103324  177777                          -1
18445  103326  077777                  3$:     77777                   ;ERROR RES.
18446  103330  177777                          -1
18447  103332  177777                          -1
18448  103334  177777                          -1
18449  103336  040200                  4$:     40200                   ;FPS BEFORE EXECUTION.
18450  103340  040206                          40206                   ;FPS AFTER EXECUTION.
18451  103342  177777                          -1                      ;FEC
18452  103344  140206                          140206                  ;ERROR FPS.
18453
18454
18455  103346  004767  000042          YYY5:   JSR     PC,STCDFS
18456  103352  177777                  1$:     177777                  ;AC0
18457  103354  177777                          -1
18458  103356  100000                          100000
18459  103360  000000                          0
18460  103362  100000                  2$:     100000                  ;RES
18461  103364  000000                          0
18462  103366  177777                          -1
18463  103370  177777                          -1
18464  103372  000000                  3$:     0                       ;ERROR RES.
18465  103374  000000                          0
18466  103376  177777                          -1
18467  103400  177777                          -1
18468  103402  047200                  4$:     47200                   ;FPS BEFORE EXECUTION.
18469  103404  147216                          147216                  ;FPS AFTER EXECUTION.
18470  103406  000010                          10                      ;FEC
18471  103410  047206                          47206                   ;ERROR FPS.
```

```
18472  103412  000460
18473
18474
18475
18476
18477
18478
18479
18480
18481
18482
18483
18484
18485
18486
18487
18488
18489
18490
18491
18492
18493
18494
18495
18496
18497
18498
18499
18500
18501
18502
18503
18504
18505  103414  012601                  STCDFS: MOV     (SP)+,R1        ;PICK UP THE POINTER TO THE OPERANDS.
18506  103416  012700  000200                  MOV     #200,R0         ;ENTER DOUBLE FLOATING MODE.
18507  103422  170100                          LDFPS   R0
18508  103424  010100                          MOV     R1,R0           ;LOAD AC0.
18509  103426  172410                          LDD     (R0),AC0
18510  103430  012700  177777                  MOV     #-1,R0          ;FILL THE OUTPUT BUFFER WITH -1'S.
18511  103434  012702  103544                  MOV     #STCDT,R2
18512  103440  012703  000004                  MOV     #4,R3
18513  103444  010022                  1$:     MOV     R0,(R2)+
18514  103446  077302                          SOB     R3,1$
18515  103450  016100  000030                  MOV     30(R1),R0       ;LOAD THE FPS.
18516  103454  170100                          LDFPS   R0
18517  103456  012700  103544                  MOV     #STCDT,R0       ;SET UP THE DESTINATION ADDRESS.
18518  103462  176010                  2$:     STCDF   AC0,(R0)        ;TEST INSTRUCTION.
18519
18520  103464  170204                          STFPS   R4              ;GET THE FPS.
18521  103466  170305                          STST    R5              ;GET THE FEC.
18522  103470  010102                          MOV     R1,R2           ;CHECK THE RESULT.
18523  103472  062702  000010                  ADD     #10,R2
18524  103476  012703  103544                  MOV     #STCDT,R3
18525  103502  012700  000004                  MOV     #4,R0
18526  103506  022223                  3$:     CMP     (R2)+,(R3)+
18527  103510  001014                          BNE     10$             ;BRANCH IF INCORRECT.
```

```
6$:     BR      YYYDONE
;THIS SUBROUTINE, STCDFS, IS USED TO SET UP THE OPERANDS, EXECUTE
;THE STCDF INSTRUCTION AND CHECK THE RESULTS. A CALL
;TO IT IS MADE THUS:
;
;               JSR     PC,@#STCFDS
;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
;               FEC:    .WORD   X               ;EXPECTED FEC
;               ERFPS:  .WORD   X               ;ERROR FPS.
;               ERR1:   ERROR   X               ;DATA ERROR.
;                       BR      CONT
;               ERR2:   ERROR   X               ;FPS ERROR.
;               CONT:                           ;RETURN ADDRESS
;
;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
;THE STCFD INSTRUCTION IS EXECUTED.
;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCFDS RETURNS CONTROL
;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCFDS
;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCFDS WILL RETURN
;TO THE ERROR CALL AT ERR2, OTHERWISE STCFDS ITSELF
;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
;STCFD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCFDS
;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCFDS WILL
;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
```

```
18528  103512  077003                          SOB     R0,3$
18529
18530  103514  016102  000032                  MOV     32(R1),R2
18531  103520  020204                          CMP     R2,R4           ;IS THE FPS CORRECT?
18532  103522  001007                          BNE     10$             ;BRANCH IF FPS INCORRECT.
18533  103524  005702                          TST     R2              ;IF EXPECTED FPS IS NEGATIVE, THEN
18534  103526  100003                          BPL     4$              ;GO AHEAD AND CHECK THE FEC.
18535  103530  026105  000034                  CMP     34(R1),R5
18536  103534  001002                          BNE     10$             ;BRANCH IF FEC IS INCORRECT.
18537  103536  000161  000040          4$:     JMP     40(R1)          ;RETURN.
18538  103542                          10$:
18539  103542  104000                          EMT                     ;
18540  103544  177777  177777  177777  STCDT:  -1,-1,-1,-1
18541  103552  177777
18542  103554                          YYYDONE:
18543  103554  004767  014420                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18544                                                                  ;SEE IF THE USER HAS EXPRESSED
18545                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18546                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18547                                                                  ;THE USER TYPED CONTROL G?).
18548                          ;*****************************************************************************
18549                          ;TEST 513       STCFD WITH ILLEGAL ACCUMULATOR TEST
18550                          ;*****************************************************************************
18551  103560                  TS513:
18552
18553  103560  012700  040000                  MOV     #40000,R0       ;DISSABLE INTERRUPTS.
18554  103564  170100                          LDFPS   R0
18555  103566  176006                  ZZZ2:   STCFD   AC0,AC6         ;THIS TEST INSTRUCTION SHOULD CAUSE AN ERROR.
18556
18557  103570  170204                          STFPS   R4              ;GET FPS.
18558  103572  170305                          STST    R5              ;GET FEC.
18559  103574  020427  140000                  CMP     R4,#140000      ;IS FPS CORRECT?
18560  103600  001004                          BNE     ZZZ10           ;BRANCH IF INCORRECT FPS.
18561  103602  022705  000002                  CMP     #2,R5           ;IS FEC CORRECT?
18562  103606  001001                          BNE     ZZZ10           ;BRANCH IF INCORRECT.
18563  103610  000401                          BR      ZZZDONE
18564
18565  103612                  ZZZ10:
18566  103612  104000                          EMT                     ;
18567
18568  103614                  ZZZDONE:
18569  103614  004767  014360                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18570                                                                  ;SEE IF THE USER HAS EXPRESSED
18571                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18572                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18573                                                                  ;THE USER TYPED CONTROL G?).
18574
18575                          ;*****************************************************************************
18576                          ;TEST 514       CLRD TEST
18577                          ;*****************************************************************************
18578  103620                  TS514:
18579  103620  012700  103724                  MOV     #AABTP1,R0      ;SET UP OUTPUT BUFFER
18580  103624  012701  103714                  MOV     #AABBF0,R1
18581  103630  012702  000004                  MOV     #4,R2
18582  103634  012021          1$:     MOV     (R0)+,(R1)+
18583  103636  077202                          SOB     R2,1$
```

```
18584  103640  012700  103714          MOV     #AABBF0,R0      ;SET UP DESTINATION OPERAND ADDRESS.
18585  103644  012701  000213          MOV     #213,R1         ;SET UP FPS.
18586  103650  170101                  LDFPS   R1
18587  103652  170410          2$:     CLRD    (R0)            ;TEST INSTRUCTION.
18588
18589  103654  170205                  STFPS   R5              ;GET FPS.
18590  103656  012702  000004          MOV     #4,R2           ;SEE IF RESULT CLEAR, 0.
18591  103662  012701  103714          MOV     #AABBF0,R1
18592  103666  005721          3$:     TST     (R1)+
18593  103670  001010                  BNE     AAB2            ;BRANCH IF RESULT INCORRECT, NOT 0.
18594  103672  077203                  SOB     R2,3$
18595  103674  022705  000204          CMP     #204,R5         ;SEE IF FPS IS CORRECT.
18596  103700  001004                  BNE     AAB2            ;BRANCH IF INCORRECT.
18597  103702  020027  103714          CMP     R0,#AABBF0      ;SEE IF R0 IS CORRECT.
18598  103706  001001                  BNE     AAB2            ;BRANCH IF R0 IS INCORRECT.
18599  103710  000411                  BR      AABDONE
18600
18601
18602  103712                  AAB2:
18603  103712  104000                  EMT                     ;
18604
18605                          ;THIS IS THE TEST DATA BUFFER, OUTPUT DATA BUFFER.
18606  103714  073475          AABBF0: 73475
18607  103716  067707                  67707
18608  103720  127347                  127347
18609  103722  056770                  56770
18610                          ;THIS IS THE DATA USED TO SET UP THE OUTPUT BUFFER.
18611  103724  073475          AABTP1: 73475
18612  103726  067707                  67707
18613  103730  127347                  127347
18614  103732  056770                  56770
18615  103734                  AABDONE:
18616  103734  004767  014240          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
18617                                                          ;SEE IF THE USER HAS EXPRESSED
18618                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
18619                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18620                                                          ;THE USER TYPED CONTROL G?).
18621
18622                          ;*****************************************************************************************
18623                          ;TEST 515       CLRD WITH ILLEGAL ACCUMULATOR TEST
18624                          ;*****************************************************************************************
18625  103740                  TS515:
18626  103740  012700  040200          MOV     #40200,R0       ;SET UP THE FPS, NO INTERRUPTS AND FD=1.
18627  103744  170100                  LDFPS   R0
18628  103746  170407          CCB2:   CLRD    AC7             ;TEST INSTRUCTION.
18629
18630  103750  170204                  STFPS   R4              ;GET FPS.
18631  103752  170305                  STST    R5              ;GET FEC.
18632  103754  020427  140200          CMP     R4,#140200              ;IS THE FPS CORRECT?
18633  103760  001004                  BNE     CCB10           ;BRANCH IF FPS IS INCORRECT.
18634  103762  022705  000002          CMP     #2,R5           ;IS THE FEC CORRECT?
18635  103766  001001                  BNE     CCB10           ;BRANCH IF FEC IS INCORRECT.
18636  103770  000401                  BR      CCBDONE
18637
18638  103772                  CCB10:
18639  103772  104000                  EMT                     ;
```

```
18640  103774
18641  103774  004767  014200          CCBDONE:
                                               JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
                                                                       ;SEE IF THE USER HAS EXPRESSED
                                                                       ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                       ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                       ;THE USER TYPED CONTROL G?).
                                        ;*****************************************************************************
                                        ;TEST 516       NEGF, ABSF AND TSTF SOURCE MODE 0 WITH ILLEGAL AC7, TEST
                                        ;*****************************************************************************
18650  104000                           TS516:
18651
18652  104000  012700  040200                  MOV     #40200,R0               ;SET UP THE FPS, FID=1 AND FD=1.
18653  104004  170100                          LDFPS   R0
18654  104006  170707                  VVB2:   NEGD    AC7             ;TEST INSTRUCTION.
18655
18656  104010  170204                          STFPS   R4              ;GET FPS.
18657  104012  170305                          STST    R5              ;GET FEC.
18658
18659  104014  022704  140200                  CMP     #140200,R4              ;IS FPS CORRECT?
18660  104020  001004                          BNE     VVB10           ;BRANCH IF FPS IS INCORRECT.
18661  104022  022705  000002                  CMP     #2,R5           ;IS FEC CORRECT?
18662  104026  001001                          BNE     VVB10           ;BRANCH IF FEC IS INCORRECT.
18663  104030  000401                          BR      VVBDONE
18664  104032                           VVB10:
18665  104032  104000                          EMT                     ;
18666
18667  104034                           VVBDONE:
18668  104034  004767  014140                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18669                                                                   ;SEE IF THE USER HAS EXPRESSED
18670                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
18671                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18672                                                                   ;THE USER TYPED CONTROL G?).
18673
18674                                   ;*****************************************************************************
18675                                   ;TEST 517       NEGF, ABSF AND TSTF SOURCE MODE 0 TEST
18676                                   ;*****************************************************************************
18677  104040                           TS517:
18678
18679  104040  012700  000200                  MOV     #200,R0         ;SET FD MODE.
18680  104044  170100                          LDFPS   R0
18681  104046  012700  104140                  MOV     #DDBTP1,R0      ;SET UP AC0.
18682  104052  172410                          LDD     (R0),AC0                ;SET AC0 = 0
18683  104054  005000                          CLR     R0              ;CLEAR THE FPS.
18684  104056  170100                          LDFPS   R0
18685  104060  012700  104150                  MOV     #DDBTP2,R0              ;LOAD AC0 TO BE A FLOATING 0.
18686  104064  172410                          LDF     (R0),AC0        ;SET AC0=ZERO
18687                                                                   ;FLOAT
18688  104066  012700  000201                  MOV     #201,R0         ;SET FD MODE.
18689  104072  170100                          LDFPS   R0
18690  104074  170700                  DDB2:   NEGD    AC0             ;TEST INSTRUCTION
18691
18692  104076  170205                          STFPS   R5              ;GET FPS.
18693  104100  012700  000200                  MOV     #200,R0         ;SET FD MODE.
18694  104104  170100                          LDFPS   R0
18695  104106  012700  104160                  MOV     #DDBBF0,R0      ;GET THE RESULT OUT OF AC0.
```

```
18696  104112  174010                  STD     AC0,(R0)
18697                                                   ;SEE IF THE RESULT IS CORRECT.
18698  104114  012701  000004          MOV     #4,R1
18699  104120  005720          1$:     TST     (R0)+
18700  104122  001005                  BNE     DDB5            ;BRANCH IF THE RESULT IS INCORRECT.
18701  104124  077103                  SOB     R1,1$
18702  104126  022705  000204          CMP     #204,R5         ;IS THE FPS CORRECT?
18703  104132  001001                  BNE     DDB5            ;BRANCH IF THE FPS IS INCORRECT.
18704  104134  000415                  BR      DDBDONE
18705  104136                  DDB5:
18706  104136  104000                  EMT                     ;
18707
18708                          ;THESE ARE TEST DATA TABLES AND AN OUTPUT BUFFER.
18709  104140  101112          DDBTP1: 101112
18710  104142  131415                  131415
18711  104144  161710                  161710
18712  104146  111213                  111213
18713  104150  000000          DDBTP2: 0
18714  104152  000000                  0
18715  104154  000000                  0
18716  104156  000000                  0
18717
18718  104160  177777          DDBBF0: -1
18719  104162  177777                  -1
18720  104164  177777                  -1
18721  104166  177777                  -1
18722
18723  104170                  DDBDONE:
18724  104170  004767  014004          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18725                                                  ;SEE IF THE USER HAS EXPRESSED
18726                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18727                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18728                                                  ;THE USER TYPED CONTROL G?).
18729
18730                          ;*****************************************************************************
18731                          ;TEST 520       NEGF, ABSF AND TSTF SOURCE MODE 1 TEST
18732                          ;*****************************************************************************
18733  104174                  TS520:
18734
18735  104174  012700  104274          MOV     #EEBTP1,R0      ;SET UP THE DATA BUFFER.
18736  104200  012701  104314          MOV     #EEBBF1,R1
18737  104204  012702  000004          MOV     #4,R2
18738  104210  012021          1$:     MOV     (R0)+,(R1)+
18739  104212  077202                  SOB     R2,1$
18740  104214  012700  000200          MOV     #200,R0         ;SET FD MODE.
18741  104220  170100                  LDFPS   R0
18742  104222  012700  104314          MOV     #EEBBF1,R0      ;SET UP THE OPERAND ADDRESS.
18743  104226  012737  104324  000004  MOV     #EEB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF ERROR.
18744  104234  170710          EEB2:   NEGD    (R0)            ;TEST INSTRUCTION.
18745
18746  104236  170205                  STFPS   R5              ;GET FPS.
18747  104240  012701  104314          MOV     #EEBBF1,R1      ;SEE IF RESULT IS CORRECT.
18748  104244  012702  000004          MOV     #4,R2
18749  104250  005721          1$:     TST     (R1)+
18750  104252  001024                  BNE     EEB10           ;BRANCH IF NOT CORRECT.
18751  104254  077203                  SOB     R2,1$
```

```
18752
18753  104256  020027  104314          CMP     R0,#EEBBF1      ;IS R0 CORRECT?
18754  104262  001020                  BNE     EEB10                   ;BRANCH IF NOT CORRECT.
18755  104264  022705  000204          CMP     #204,R5         ;IS THE FPS CORRECT?
18756  104270  001015                  BNE     EEB10           ;BRANCH IF NOT CORRECT.
18757  104272  000415                  BR      EEBDONE
18758
18759                                  ;THESE ARE TEST DATA TABLES AND A BUFFER.
18760  104274  000177          EEBTP1: 177
18761  104276  167574                  167574
18762  104300  137271                  137271
18763  104302  107675                  107675
18764  104304  177777          EEBBF0: -1
18765  104306  177777                  -1
18766  104310  177777                  -1
18767  104312  177777                  -1
18768  104314  177777          EEBBF1: -1
18769  104316  177777                  -1
18770  104320  177777                  -1
18771  104322  177777                  -1
18772  104324                  EEB10:
18773  104324  104000                  EMT                     ;
18774  104326                  EEBDONE:
18775  104326  004767  013646          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
18776                                                          ;SEE IF THE USER HAS EXPRESSED
18777                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
18778                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18779                                                          ;THE USER TYPED CONTROL G?).
18780
18781                                  ;*****************************************************************************
18782                                  ;TEST 521       NEGF, ABSF AND TSTF SOURCE MODE 2 TEST
18783                                  ;*****************************************************************************
18784  104332                  TS521:
18785
18786  104332  012700  104432                  MOV     #FFBTP1,R0              ;SET UP THE DATA BUFFER.
18787  104336  012701  104442                  MOV     #FFBBF1,R1
18788  104342  012702  000004                  MOV     #4,R2
18789  104346  012021          1$:     MOV     (R0)+,(R1)+
18790  104350  077202                  SOB     R2,1$
18791  104352  012700  000200                  MOV     #200,R0         ;SET FD.
18792  104356  170100                  LDFPS   R0
18793  104360  012700  104442                  MOV     #FFBBF1,R0      ;SET UP THE OPERAND ADDRESS.
18794  104364  012737  104452  000004        MOV     #FFB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
18795
18796  104372  170620          FFB2:   ABSD    (R0)+           ;TEST INSTRUCTION.
18797
18798  104374  170205                  STFPS   R5              ;GET FPS.
18799  104376  012701  104442                  MOV     #FFBBF1,R1              ;CHECK RESULT.
18800  104402  012702  000004                  MOV     #4,R2
18801  104406  005721          1$:     TST     (R1)+
18802  104410  001020                  BNE     FFB10           ;BRANCH IF INCORRECT.
18803  104412  077203                  SOB     R2,1$
18804
18805  104414  020027  104452                  CMP     R0,#FFBBF1+10   ;IS R0 CORRECT?
18806  104420  001014                  BNE     FFB10           ;BRANCH IF INCORRECT.
18807  104422  022705  000204                  CMP     #204,R5         ;IS THE FPS CORRECT?
```

```
18808  104426  001011                  BNE     FFB10           ;BRANCH IF INCORRECT.
18809  104430  000411                  BR      FFBDONE
18810
18811                          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
18812  104432  000177            FFBTP1: 177
18813  104434  167574                    167574
18814  104436  137271                    137271
18815  104440  107675                    107675
18816  104442  177777            FFBBF1: -1
18817  104444  177777                    -1
18818  104446  177777                    -1
18819  104450  177777                    -1
18820  104452                    FFB10:
18821  104452  104000                    EMT                     ;
18822  104454                    FFBDONE:
18823  104454  004767  013520            JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
18824                                                            ;SEE IF THE USER HAS EXPRESSED
18825                                                            ;THE DESIRE TO CHANGE THE SOFTWARE
18826                                                            ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18827                                                            ;THE USER TYPED CONTROL G?).
18828                          ;*****************************************************************************
18829                          ;TEST 522       NEGF, ABSF AND TSTF SOURCE MODE 4 TEST
18830                          ;*****************************************************************************
18831  104460                  TS522:
```

```
18832
18833  104460  012700  104560          MOV     #GGBTP1,R0              ;SET UP THE DATA BUFFER.
18834  104464  012701  104570          MOV     #GGBBF0,R1
18835  104470  012702  000004          MOV     #4,R2
18836  104474  012021          1$:     MOV     (R0)+,(R1)+
18837  104476  077202                  SOB     R2,1$
18838  104500  012700  000200          MOV     #200,R0         ;SET FD.
18839  104504  170100                  LDFPS   R0
18840  104506  012700  104600          MOV     #GGBBF1,R0      ;SET UP THE OPERAND ADDRESS.
18841  104512  012737  104610  000004  MOV     #GGB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
18842
18843  104520  170640          GGB2:   ABSD    -(R0)           ;TEST INSTRUCTION.
18844
18845  104522  170205                  STFPS   R5              ;GET FPS.
18846  104524  012701  104570          MOV     #GGBBF0,R1              ;CHECK RESULT.
18847  104530  012702  000004          MOV     #4,R2
18848  104534  005721          1$:     TST     (R1)+
18849  104536  001024                  BNE     GGB10           ;BRANCH IF INCORRECT.
18850  104540  077203                  SOB     R2,1$
18851
18852  104542  020027  104570          CMP     R0,#GGBBF0      ;IS R0 CORRECT?
18853  104546  001020                  BNE     GGB10           ;BRANCH IF INCORRECT.
18854  104550  022705  000204          CMP     #204,R5         ;IS THE FPS CORRECT?
18855  104554  001015                  BNE     GGB10           ;BRANCH IF INCORRECT.
18856  104556  000415                  BR      GGBDONE
18857
18858                          ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
18859  104560  000177          GGBTP1: 177
18860  104562  117273                  117273
18861  104564  147576                  147576
18862  104566  177071                  177071
18863  104570  177777          GGBBF0: -1
18864  104572  177777                  -1
18865  104574  177777                  -1
18866  104576  177777                  -1
18867  104600  177777          GGBBF1: -1
18868  104602  177777                  -1
18869  104604  177777                  -1
18870  104606  177777                  -1
18871  104610                  GGB10:
18872  104610  104000                  EMT                     ;
18873  104612                  GGBDONE:
18874  104612  004767  013362          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18875                                                          ;SEE IF THE USER HAS EXPRESSED
18876                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
18877                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18878                                                          ;THE USER TYPED CONTROL G?).
18879                          ;*****************************************************************************
18880                          ;TEST 523       NEGF, ABSF AND TSTF SOURCE MODE 3 TEST
18881                          ;************** *****************************************************************
18882  104616                  TS523:
18883
18884  104616  012700  104716          MOV     #HHBTP1,R0              ;SET UP THE DATA BUFFER.
18885  104622  012701  104736          MOV     #HHBBF0,R1
18886  104626  012702  000010          MOV     #10,R2
18887  104632  012021          1$:     MOV     (R0)+,(R1)+
```

```
18888  104634  077202                          SOB     R2,1$
18889  104636  012700  000200                  MOV     #200,R0         ;SET FD.
18890  104642  170100                          LDFPS   R0
18891  104644  012700  104746                  MOV     #HHBBF1,R0      ;SET UP THE OPERAND ADDRESS.
18892  104650  012737  104756  000004          MOV     #HHB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
18893
18894  104656  170630                  HHB2:   ABSD    @(R0)+          ;TEST INSTRUCTION.
18895
18896  104660  170205                          STFPS   R5              ;GET FPS.
18897  104662  012701  104736                  MOV     #HHBBF0,R1              ;CHECK RESULT.
18898  104666  012702  000004                  MOV     #4,R2
18899  104672  005721                  1$:     TST     (R1)+
18900  104674  001030                          BNE     HHB10           ;BRANCH IF INCORRECT.
18901  104676  077203                          SOB     R2,1$
18902  104700  020027  104750                  CMP     R0,#HHBBF1+2    ;IS R0 CORRECT?
18903  104704  001024                          BNE     HHB10           ;BRANCH IF INCORRECT.
18904  104706  022705  000204                  CMP     #204,R5         ;IS THE FPS CORRECT?
18905  104712  001021                          BNE     HHB10           ;BRANCH IF INCORRECT.
18906  104714  000421                          BR      HHBDONE
18907
18908                                  ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
18909  104716  000177                  HHBTP1: 177
18910  104720  147576                          147576
18911  104722  177071                          177071
18912  104724  107576  104736  177777          107576,HHBBF0,-1,-1,-1
18913  104732  177777  177777
18914  104736  177777                  HHBBF0: -1
18915  104740  177777                          -1
18916  104742  177777                          -1
18917  104744  177777                          -1
18918  104746  177777                  HHBBF1: -1
18919  104750  177777                          -1
18920  104752  177777                          -1
18921  104754  177777                          -1
18922  104756                          HHB10:
18923  104756  104000                          EMT                     ;
18924  104760                          HHBDONE:
18925  104760  004767  013214                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18926                                                                  ;SEE IF THE USER HAS EXPRESSED
18927                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18928                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18929                                                                  ;THE USER TYPED CONTROL G?).
18930                                  ;*****************************************************************************
18931                                  ;TEST 524       NEGF, ABSF AND TSTF SOURCE MODE 5 TEST
18932                                  ;*****************************************************************************
18933  104764                          TS524:
18934
18935  104764  012700  105064                  MOV     #IIBTP1,R0              ;SET UP THE DATA BUFFER.
18936  104770  012701  105104                  MOV     #IIBBF0,R1
18937  104774  012702  000010                  MOV     #10,R2
18938  105000  012021                  1$:     MOV     (R0)+,(R1)+
18939  105002  077202                          SOB     R2,1$
18940  105004  012700  000200                  MOV     #200,R0         ;SET FD.
18941  105010  170100                          LDFPS   R0
18942  105012  012700  105116                  MOV     #IIBBF1+2,R0    ;SET UP THE OPERAND ADDRESS.
18943  105016  012737  105124  000004          MOV     #IIB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
```

```
18944
18945  105024  170750                  IIB2:   NEGD    @-(R0)          ;TEST INSTRUCTION.
18946
18947  105026  170205                          STFPS   R5              ;GET FPS.
18948  105030  012701  105104                  MOV     #IIBBF0,R1              ;CHECK RESULT.
18949  105034  012702  000004                  MOV     #4,R2
18950  105040  005721                  1$:     TST     (R1)+
18951  105042  001030                          BNE     IIB10           ;BRANCH IF INCORRECT.
18952  105044  077203                          SOB     R2,1$
18953  105046  020027  105114                  CMP     R0,#IIBBF1      ;IS R0 CORRECT?
18954  105052  001024                          BNE     IIB10           ;BRANCH IF INCORRECT.
18955  105054  022705  000204                  CMP     #204,R5         ;IS THE FPS CORRECT?
18956  105060  001021                          BNE     IIB10           ;BRANCH IF INCORRECT.
18957  105062  000421                          BR      IIBDONE
18958
18959                                  ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
18960  105064  000176                  IIBTP1: 176
18961  105066  177074                          177074
18962  105070  127374                          127374
18963  105072  157677  105104  177777        157677,IIBBF0,-1,-1,-1
18964  105100  177777  177777
18965  105104  177777                  IIBBF0: -1
18966  105106  177777                          -1
18967  105110  177777                          -1
18968  105112  177777                          -1
18969  105114  177777                  IIBBF1: -1
18970  105116  177777                          -1
18971  105120  177777                          -1
18972  105122  177777                          -1
18973
18974  105124                          IIB10:
18975  105124  104000                          EMT                     ;
18976  105126                          IIBDONE:
18977  105126  004767  013046                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
18978                                                                  ;SEE IF THE USER HAS EXPRESSED
18979                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
18980                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
18981                                                                  ;THE USER TYPED CONTROL G?).
18982                                  ;*****************************************************************************
18983                                  ;TEST 525       NEGF, ABSF AND TSTF SOURCE MODE 6 TEST
18984                                  ;*****************************************************************************
18985  105132                          TS525:
18986
18987  105132  012700  105234                  MOV     #JJBTP1,R0              ;SET UP THE DATA BUFFER.
18988  105136  012701  105246                  MOV     #JJBBF0,R1
18989  105142  012702  000004                  MOV     #4,R2
18990  105146  012021                  1$:     MOV     (R0)+,(R1)+
18991  105150  077202                          SOB     R2,1$
18992  105152  012700  000200                  MOV     #200,R0         ;SET FD.
18993  105156  170100                          LDFPS   R0
18994  105160  012700  105237                  MOV     #JJBBF0-7,R0    ;SET UP THE OPERAND ADDRESS.
18995  105164  012737  105256  000004          MOV     #JJB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
18996
18997  105172  170660  000007          JJB2:   ABSD    7(R0)           ;TEST INSTRUCTION.
18998
18999  105176  170205                          STFPS   R5              ;GET FPS.
```

```
19000  105200  012701  105246          MOV     #JJBBF0,R1              ;CHECK RESULT.
19001  105204  012702  000004          MOV     #4,R2
19002  105210  005721          1$:     TST     (R1)+
19003  105212  001021                  BNE     JJB10           ;BRANCH IF INCORRECT.
19004  105214  077203                  SOB     R2,1$
19005  105216  020027  105237          CMP     R0,#JJBBF0-7    ;IS R0 CORRECT?
19006  105222  001015                  BNE     JJB10           ;BRANCH IF INCORRECT.
19007  105224  022705  000204          CMP     #204,R5         ;IS THE FPS CORRECT?
19008  105230  001012                  BNE     JJB10           ;BRANCH IF INCORRECT.
19009  105232  000412                  BR      JJ3DONE
19010
19011                                  ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
19012  105234  000177          JJBTP1: 177
19013  105236  161524                  161524
19014  105240  131273                  131273
19015  105242  107174  000000          107174,
19016  105246  177777          JJBBF0: -1
19017  105250  177777                  -1
19018  105252  177777                  -1
19019  105254  177777                  -1
19020  105256                  JJB10:
19021  105256  104000                  EMT                     ;
19022  105260                  JJBDONE:
19023  105260  004767  012714          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
19024                                                          ;SEE IF THE USER HAS EXPRESSED
19025                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
19026                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19027                                                          ;THE USER TYPED CONTROL G?).
19028                                  ;*****************************************************************************
19029                                  ;TEST 526       NEGF, ABSF AND TSTF SOURCE MODE 7 TEST
19030                                  ;*****************************************************************************
19031  105264                  TS526:
19032
19033  105264  012700  105366          MOV     #KKBTP1,R0              ;SET UP THE DATA BUFFER.
19034  105270  012701  105406          MOV     #KKBBF0,R1
19035  105274  012702  000010          MOV     #10,R2
19036  105300  012021          1$:     MOV     (R0)+,(R1)+
19037  105302  077202                  SOB     R2,1$
19038  105304  012700  000200          MOV     #200,R0         ;SET FD.
19039  105310  170100                  LDFPS   R0
19040  105312  012700  105407          MOV     #KKBBF1-7,R0    ;SET UP THE OPERAND ADDRESS.
19041  105316  012737  105426  000004  MOV     #KKB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
19042
19043  105324  170770  000007  KKB2:   NEGD    @7(R0)          ;TEST INSTRUCTION.
19044
19045  105330  170205                  STFPS   R5              ;GET FPS.
19046  105332  012701  105406          MOV     #KKBBF0,R1              ;CHECK RESULT.
19047  105336  012702  000004          MOV     #4,R2
19048  105342  005721          1$:     TST     (R1)+
19049  105344  001030                  BNE     KKB10           ;BRANCH IF INCORRECT.
19050  105346  077203                  SOB     R2,1$
19051  105350  020027  105407          CMP     R0,#KKBBF1-7    ;IS R0 CORRECT?
19052  105354  001024                  BNE     KKB10           ;BRANCH IF INCORRECT.
19053  105356  022705  000204          CMP     #204,R5         ;IS THE FPS CORRECT?
19054  105362  001021                  BNE     KKB10           ;BRANCH IF INCORRECT.
19055  105364  000421                  BR      KKBDONE
```

```
19056
19057                                   ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
19058  105366  000177                   KKBTP1: 177
19059  105370  167574                           167574
19060  105372  137271                           137271
19061  105374  107675  105406  177777          107675,KKBBF0,-1,-1,-1
19062  105402  177777  177777
19063  105406  177777                   KKBBF0: -1
19064  105410  177777                           -1
19065  105412  177777                           -1
19066  105414  177777                           -1
19067  105416  177777                   KKBBF1: -1
19068  105420  177777                           -1
19069  105422  177777                           -1
19070  105424  177777                           -1
19071  105426                           KKB10:
19072  105426  104000                           EMT                     ;
19073  105430                           KKBDONE:
19074  105430  004767  012544                   JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19075                                                                   ;SEE IF THE USER HAS EXPRESSED
19076                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
19077                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19078                                                                   ;THE USER TYPED CONTROL G?).
19079                                   ;*****************************************************************************
19080                                   ;TEST 527       NEGF, ABSF AND TSTF SOURCE MODE 6, GR7, TEST
19081                                   ;*****************************************************************************
19082  105434                           TS527:
19083  105434  012700  105524                   MOV     #LLBTP1,R0              ;SET UP THE DATA BUFFER.
19084  105440  012701  105534                   MOV     #LLBBF0,R1
19085  105444  012702  000004                   MOV     #4,R2
19086  105450  012021                   1$:     MOV     (R0)+,(R1)+
19087  105452  077202                           SOB     R2,1$
19088  105454  012700  000200                   MOV     #200,R0         ;SET FD.
19089  105460  170100                           LDFPS   R0
19090  105462  012737  105544  000004           MOV     #LLB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
19091
19092  105470  170767  000040           LLB2:   NEGD    LLBBF0          ;TEST INSTRUCTION.
19093
19094  105474  170205                           STFPS   R5              ;GET FPS.
19095  105476  012701  105534                   MOV     #LLBBF0,R1              ;CHECK RESULT.
19096  105502  012702  000004                   MOV     #4,R2
19097  105506  005721                   1$:     TST     (R1)+
19098  105510  001015                           BNE     LLB10           ;BRANCH IF INCORRECT.
19099  105512  077203                           SOB     R2,1$
19100  105514  022705  000204                   CMP     #204,R5         ;IS THE FPS CORRECT?
19101  105520  001011                           BNE     LLB10           ;BRANCH IF INCORRECT.
19102  105522  000411                           BR      LLBDONE
19103
19104                                   ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
19105  105524  000127                   LLBTP1: 127
19106  105526  137475                           137475
19107  105530  147372                           147372
19108  105532  117057                           117057
19109  105534  177777                   LLBBF0: -1
19110  105536  177777                           -1
19111  105540  177777                           -1
```

```
19112  105542  177777                          -1
19113
19114  105544                          LLB10:
19115  105544  104000                          EMT                     ;
19116  105546                          LLBDONE:
19117  105546  004767  012426                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19118                                                                  ;SEE IF THE USER HAS EXPRESSED
19119                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19120                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19121                                                                  ;THE USER TYPED CONTROL G?).
19122                                   ;*****************************************************************************
19123                                   ;TEST 530       NEGF, ABSF AND TSTF SOURCE MODE 7, GR7, TEST
19124                                   ;*****************************************************************************
19125  105552                          TS530:
19126
19127  105552  012700  105642                  MOV     #MMBTP1,R0              ;SET UP THE DATA BUFFER.
19128  105556  012701  105662                  MOV     #MMBBF0,R1
19129  105562  012702  000010                  MOV     #10,R2
19130  105566  012021                  1$:     MOV     (R0)+,(R1)+
19131  105570  077202                          SOB     R2,1$
19132  105572  012700  000200                  MOV     #200,R0         ;SET FD.
19133  105576  170100                          LDFPS   R0
19134  105600  012737  105702  000004          MOV     #MMB10,@#ERRVECT ;SET UP VECTOR 4 IN CASE OF AN ERROR.
19135
19136  105606  170677  000060          MMB2:   ABSD    @MMBBF1         ;TEST INSTRUCTION.
19137
19138  105612  170205                          STFPS   R5              ;GET FPS.
19139  105614  012701  105662                  MOV     #MMBBF0,R1              ;CHECK RESULT.
19140  105620  012702  000004                  MOV     #4,R2
19141  105624  005721                  1$:     TST     (R1)+
19142  105626  001025                          BNE     MMB10           ;BRANCH IF INCORRECT.
19143  105630  077203                          SOB     R2,1$
19144  105632  022705  000204                  CMP     #204,R5         ;IS THE FPS CORRECT?
19145  105636  001021                          BNE     MMB10           ;BRANCH IF INCORRECT.
19146  105640  000421                          BR      MMBDONE
19147
19148                                   ;THESE ARE TEST DATA TABLES AND DATA BUFFER.
19149  105642  000137                  MMBTP1: 137
19150  105644  045607                          045607
19151  105646  101230                          101230
19152  105650  045607  105662  177777          45607,MMBBF0,-1,-1,-1
19153  105656  177777  177777
19154  105662  177777                  MMBBF0: -1
19155  105664  177777                          -1
19156  105666  177777                          -1
19157  105670  177777                          -1
19158  105672  177777                  MMBBF1: -1
19159  105674  177777                          -1
19160  105676  177777                          -1
19161  105700  177777                          -1
19162
19163  105702                          MMB10:
19164  105702  104000                          EMT                     ;
19165  105704                          MMBDONE:
19166  105704  004767  012270                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19167                                                                  ;SEE IF THE USER HAS EXPRESSED
```

```
19168                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
19169                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19170                                                   ;THE USER TYPED CONTROL G?).
19171                   ;*****************************************************************************
19172                   ;TEST 531       SPECIAL DEST, MODE 0, TEST
19173                   ;*****************************************************************************
19174  105710           TS531:
19175
19176  105710  012700  000200           MOV     #200,R0         ;SET FD.
19177  105714  170100                   LDFPS   R0
19178  105716  012700  105776           MOV     #NNBTP1,R0              ;SET UP AC0.
19179  105722  172410                   LDD     (R0),AC0
19180  105724  170700           NNB2:   NEGD    AC0             ;TEST INSTRUCTION.
19181
19182  105726  170205                   STFPS   R5              ;GET FPS.
19183  105730  012700  000200           MOV     #200,R0         ;SET FD.
19184  105734  170100                   LDFPS   R0
19185  105736  012700  106016           MOV     #NNBBF0,R0              ;GET THE RESULT.
19186  105742  174010                   STD     AC0,(R0)
19187  105744  012700  106016           MOV     #NNBBF0,R0              ;IS THE RESULT CORRECT?
19188  105750  012701  106006           MOV     #NNBTP2,R1
19189  105754  012702  000004           MOV     #4,R2
19190  105760  022021           1$:     CMP     (R0)+,(R1)+
19191  105762  001021                   BNE     NNB10           ;BRANCH IF INCORRECT.
19192  105764  077203                   SOB     R2,1$
19193  105766  022705  000210           CMP     #210,R5         ;IS THE FPS CORRECT?
19194  105772  001015                   BNE     NNB10           ;BRANCH IF INCORRECT.
19195  105774  000415                   BR      NNBDONE
19196
19197                                   ;THESE ARE DATA TABLES AND A DATA BUFFER.
19198  105776  013572           NNBTP1: 013572
19199  106000  046013                   46013
19200  106002  057246                   57246
19201  106004  013570                   013570
19202  106006  113572           NNBTP2: 113572
19203  106010  046013                   46013
19204  106012  057246                   57246
19205  106014  013570                   013570
19206  106016  000000           NNBBF0: 0
19207  106020  000000                   0
19208  106022  000000                   0
19209  106024  000000                   0
19210
19211  106026                   NNB10:
19212  106026  104000                   EMT                     ;
19213  106030           NNBDONE:
19214  106030  004767  012144           JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
19215                                                           ;SEE IF THE USER HAS EXPRESSED
19216                                                           ;THE DESIRE TO CHANGE THE SOFTWARE
19217                                                           ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19218                                                           ;THE USER TYPED CONTROL G?).
19219                   ;*****************************************************************************
19220                   ;TEST 532       SPECIAL DEST, MODE 1, TEST
19221                   ;*****************************************************************************
19222  106034           TS532:
19223
```

```
19224  106034  012701  106136          MOV     #OOBTP1,R1      ;SET UP THE DATA BUFFER.
19225  106040  012700  106146          MOV     #OOBTP2,R0
19226  106044  012702  000004          MOV     #4,R2
19227  106050  012021                  1$:     MOV     (R0)+,(R1)+
19228  106052  077202                          SOB     R2,1$
19229  106054  012700  106136          MOV     #OOBTP1,R0
19230  106060  042710  100000          BIC     #100000,(R0)    ;MAKE OPERAND POSITIVE.
19231  106064  012701  000200          MOV     #200,R1         ;SET FD.
19232  106070  170101                  LDFPS   R1
19233
19234  106072  170710                  OOB2:   NEGD    (R0)            ;TEST INSTRUCTION.
19235  106074  170205                  STFPS   R5              ;GET FPS.
19236  106076  012701  106136          MOV     #OOBTP1,R1              ;IS THE RESULT CORRECT.
19237  106102  012702  106146          MOV     #OOBTP2,R2
19238  106106  012703  000004          MOV     #4,R3
19239  106112  022122                  1$:     CMP     (R1)+,(R2)+
19240  106114  001020                  BNE     OOB10           ;BRANCH IF INCORRECT.
19241  106116  077303                  SOB     R3,1$
19242  106120  022700  106136          CMP     #OOBTP1,R0      ;IS R0 CORRECT.
19243  106124  001014                  BNE     OOB10           ;BRANCH IF INCORRECT.
19244  106126  022705  000210          CMP     #210,R5         ;IS THE FPS CORRECT?
19245  106132  001011                  BNE     OOB10           ;BRANCH IF INCORRECT.
19246  106134  000411                  BR      OOBDONE
19247
19248                                  ;THESE ARE DATA TABLES AND A DATA BUFFER.
19249  106136  023245                  OOBTP1: 023245
19250  106140  026720                          26720
19251  106142  122324                          122324
19252  106144  052672                          52672
19253  106146  123245                  OOBTP2: 123245
19254  106150  026720                          26720
19255  106152  122324                          122324
19256  106154  052672                          52672
19257
19258  106156                          OOB10:
19259  106156  104000                          EMT             ;
19260  106160                          OOBDONE:
19261  106160  004767  012014                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19262                                                                  ;SEE IF THE USER HAS EXPRESSED
19263                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19264                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19265                                                                  ;THE USER TYPED CONTROL G?).
19266                                  ;*****************************************************************************
19267                                  ;TEST 533       SPECIAL DEST, MODE 2, TEST
19268                                  ;*****************************************************************************
19269  106164                          TS533:
19270
19271  106164  012701  106266          MOV     #PPBTP1,R1      ;SET UP THE DATA BUFFER.
19272  106170  012700  106276          MOV     #PPBTP2,R0
19273  106174  012702  000004          MOV     #4,R2
19274  106200  012021                  1$:     MOV     (R0)+,(R1)+
19275  106202  077202                          SOB     R2,1$
19276  106204  012700  106266          MOV     #PPBTP1,R0
19277  106210  042710  100000          BIC     #100000,(R0)    ;MAKE OPERAND POSITIVE.
19278  106214  012701  000200          MOV     #200,R1         ;SET FD.
19279  106220  170101                  LDFPS   R1
```

```
19280
19281  106222  170720                  PPB2:   NEGD    (R0)+           ;TEST INSTRUCTION.
19282
19283  106224  170205                          STFPS   R5              ;GET FPS.
19284  106226  012701  106266                  MOV     #PPBTP1,R1              ;IS THE RESULT CORRECT.
19285  106232  012702  106276                  MOV     #PPBTP2,R2
19286  106236  012703  000004                  MOV     #4,R3
19287  106242  022122                  1$:     CMP     (R1)+,(R2)+
19288  106244  001020                          BNE     PPB10           ;BRANCH IF INCORRECT.
19289  106246  077303                          SOB     R3,1$
19290  106250  022700  106276                  CMP     #PPBTP1+10,R0   ;IS R0 CORRECT.
19291  106254  001014                          BNE     PPB10           ;BRANCH IF INCORRECT.
19292  106256  022705  000210                  CMP     #210,R5         ;IS THE FPS CORRECT?
19293  106262  001011                          BNE     PPB10           ;BRANCH IF INCORRECT.
19294  106264  000411                          BR      PPBDONE
19295
19296                                  ;THESE ARE DATA TABLES AND A DATA BUFFER.
19297  106266  023245                  PPBTP1: 023245
19298  106270  026720                          26720
19299  106272  122324                          122324
19300  106274  052672                          52672
19301  106276  123245                  PPBTP2: 123245
19302  106300  026720                          26720
19303  106302  122324                          122324
19304  106304  052672                          52672
19305
19306  106306                          PPB10:
19307  106306  104000                          EMT                     ;
19308  106310                          PPBDONE:
19309  106310  004767  011664                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19310                                                                  ;SEE IF THE USER HAS EXPRESSED
19311                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19312                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19313                                                                  ;THE USER TYPED CONTROL G?).
19314          ;*****************************************************************************************
19315          ;TEST 534       SPECIAL DEST, MODE 4, TEST
19316          ;*****************************************************************************************
19317  106314                  TS534:
19318  106314  012701  106420                  MOV     #QQBTP1,R1      ;SET UP THE DATA BUFFER.
19319  106320  012700  106440                  MOV     #QQBTP2,R0
19320  106324  012702  000004                  MOV     #4,R2
19321  106330  012021                  1$:     MOV     (R0)+,(R1)+
19322  106332  077202                          SOB     R2,1$
19323  106334  012700  106430                  MOV     #QQBTP1+10,R0
19324  106340  042760  100000  177770          BIC     #100000,-10(R0) ;MAKE OPERAND POSITIVE.
19325  106346  012701  000200                  MOV     #200,R1         ;SET FD.
19326  106352  170101                          LDFPS   R1
19327
19328  106354  170740                  QQB2:   NEGD    -(R0)           ;TEST INSTRUCTION.
19329
19330  106356  170205                          STFPS   R5              ;GET FPS.
19331  106360  012701  106420                  MOV     #QQBTP1,R1              ;IS THE RESULT CORRECT.
19332  106364  012702  106440                  MOV     #QQBTP2,R2
19333  106370  012703  000004                  MOV     #4,R3
19334  106374  022122                  1$:     CMP     (R1)+,(R2)+
19335  106376  001024                          BNE     QQB10           ;BRANCH IF INCORRECT.
```

```
19336  106400  077303                          SOB     R3,1$
19337  106402  022700  106420                  CMP     #QQBTP1,R0      ;IS R0 CORRECT.
19338  106406  001020                          BNE     QQB10           ;BRANCH IF INCORRECT.
19339  106410  022705  000210                  CMP     #210,R5         ;IS THE FPS CORRECT?
19340  106414  001015                          BNE     QQB10           ;BRANCH IF INCORRECT.
19341  106416  000415                          BR      QQBDONE
19342
19343                                  ;THESE ARE DATA TABLES AND A DATA BUFFER.
19344  106420  023245                  QQBTP1: 023245
19345  106422  026720                          26720
19346  106424  122324                          122324
19347  106426  052672                          52672
19348  106430  177777  177777  177777          .WORD   -1,-1,-1,-1
19349  106436  177777
19350  106440  123245                  QQBTP2: 123245
19351  106442  026720                          26720
19352  106444  122324                          122324
19353  106446  052672                          52672
19354
19355  106450                          QQB10:
19356  106450  104000                          EMT                     ;
19357  106452                          QQBDONE:
19358  106452  004767  011522                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
19359                                                                  ;SEE IF THE USER HAS EXPRESSED
19360                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19361                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19362                                                                  ;THE USER TYPED CONTROL G?).
19363
19364                                  ;*****************************************************************************
19365                                  ;TEST 535       SPECIAL DEST, MODE 3, TEST
19366                                  ;*****************************************************************************
19367  106456                          TS535:
19368
19369  106456  012701  106566                  MOV     #RRBTP1,R1      ;SET UP THE DATA BUFFER.
19370  106462  012700  106576                  MOV     #RRBTP2,R0
19371  106466  012702  000004                  MOV     #4,R2
19372  106472  012021                  1$:     MOV     (R0)+,(R1)+
19373  106474  077202                          SOB     R2,1$
19374  106476  012700  106606                  MOV     #RRBTP3,R0
19375  106502  012710  106566                  MOV     #RRBTP1,(R0)
19376  106506  042737  100000  106566          BIC     #100000,@#RRBTP1        ;MAKE THE OPERAND POSITIVE.
19377  106514  012701  000200                  MOV     #200,R1         ;SET FD.
19378  106520  170101                          LDFPS   R1
19379
19380  106522  170730                  RRB2:   NEGD    @(R0)+          ;TEST INSTRUCTION.
19381
19382  106524  170205                          STFPS   R5              ;GET FPS.
19383  106526  012701  106566                  MOV     #RRBTP1,R1              ;IS THE RESULT CORRECT.
19384  106532  012702  106576                  MOV     #RRBTP2,R2
19385  106536  012703  000004                  MOV     #4,R3
19386  106542  022122                  1$:     CMP     (R1)+,(R2)+
19387  106544  001021                          BNE     RRB10           ;BRANCH IF INCORRECT.
19388  106546  077303                          SOB     R3,1$
19389  106550  022700  106610                  CMP     #RRBTP3+2,R0    ;IS R0 CORRECT.
19390  106554  001015                          BNE     RRB10           ;BRANCH IF INCORRECT.
19391  106556  022705  000210                  CMP     #210,R5         ;IS THE FPS CORRECT?
```

```
19392  106562  001012                    BNE     RRB10           ;BRANCH IF INCORRECT.
19393  106564  000412                    BR      RRBDONE
19394
19395                                    ;THESE ARE DATA TABLES AND A DATA BUFFER.
19396  106566  023245                    RRBTP1: 023245
19397  106570  026720                            26720
19398  106572  122324                            122324
19399  106574  052672                            52672
19400  106576  123245                    RRBTP2: 123245
19401  106600  026720                            26720
19402  106602  123324                            123324
19403  106604  052672                            52672
19404  106606  106566                    RRBTP3: RRBTP1
19405
19406  106610                            RRB10:
19407  106610  104000                            EMT                     ;
19408  106612                            RRBDONE:
19409  106612  004767  011362                    JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19410                                                                    ;SEE IF THE USER HAS EXPRESSED
19411                                                                    ;THE DESIRE TO CHANGE THE SOFTWARE
19412                                                                    ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19413                                                                    ;THE USER TYPED CONTROL G?).
19414
19415                                    ;*****************************************************************************
19416                                    ;TEST 536       SPECIAL DEST, MODE 5, TEST
19417                                    ;*****************************************************************************
19418  106616                            TS536:
19419  106616  012701  106730                    MOV     #SSBTP1,R1      ;SET UP THE DATA BUFFER.
19420  106622  012700  106740                    MOV     #SSBTP2,R0
19421  106626  012702  000004                    MOV     #4,R2
19422  106632  012021                    1$:     MOV     (R0)+,(R1)+
19423  106634  077202                            SOB     R2,1$
19424  106636  012700  106752                    MOV     #SSBTP3+2,R0
19425  106642  012760  106730  177776          MOV     #SSBTP1,-2(R0)
19426  106650  042737  100000  106730          BIC     #100000,@#SSBTP1        ;MAKE THE OPERAND POSITIVE.
19427  106656  012701  000200                    MOV     #200,R1         ;SET FD.
19428  106662  170101                            LDFPS   R1
19429
19430  106664  170750                    SSB2:   NEGD    @-(R0)          ;TEST INSTRUCTION.
19431
19432  106666  170205                            STFPS   R5              ;GET FPS.
19433  106670  012701  106730                    MOV     #SSBTP1,R1              ;IS THE RESULT CORRECT.
19434  106674  012702  106740                    MOV     #SSBTP2,R2
19435  106700  012703  000004                    MOV     #4,R3
19436  106704  022122                    1$:     CMP     (R1)+,(R2)+
19437  106706  001021                            BNE     SSB10           ;BRANCH IF INCORRECT.
19438  106710  077303                            SOB     R3,1$
19439  106712  022700  106750                    CMP     #SSBTP3,R0      ;IS R0 CORRECT.
19440  106716  001015                            BNE     SSB10           ;BRANCH IF INCORRECT.
19441  106720  022705  000210                    CMP     #210,R5         ;IS THE FPS CORRECT?
19442  106724  001012                            BNE     SSB10           ;BRANCH IF INCORRECT.
19443  106726  000412                            BR      SSBDONE
19444
19445                                    ;THESE ARE DATA TABLES AND A DATA BUFFER.
19446  106730  023245                    SSBTP1: 023245
19447  106732  026720                            26720
```

```
19448  106734  122324                  122324
19449  106736  052672                  52672
19450  106740  123245          SSBTP2: 123245
19451  106742  026270                  26270
19452  106744  122324                  122324
19453  106746  052672                  52672
19454  106750  106730          SSBTP3: SSBTP1
19455
19456  106752                  SSB10:
19457  106752  104000                  EMT                     ;
19458  106754                  SSBDONE:
19459  106754  004767  011220          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
19460                                                          ;SEE IF THE USER HAS EXPRESSED
19461                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
19462                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19463                                                          ;THE USER TYPED CONTROL G?).
19464          ;*****************************************************************************
19465          ;TEST 537       SPECIAL DEST, FLOATING MODE 2, TEST
19466          ;*****************************************************************************
19467  106760                  TS537:
19468  106760  012701  107062          MOV     #TTBTP1,R1      ;SET UP THE DATA BUFFER.
19469  106764  012700  107072          MOV     #TTBTP2,R0
19470  106770  012702  000004          MOV     #4,R2
19471  106774  012021          1$:     MOV     (R0)+,(R1)+
19472  106776  077202                  SOB     R2,1$
19473  107000  012700  107062          MOV     #TTBTP1,R0
19474  107004  042710  100000          BIC     #100000,(R0)    ;MAKE OPERAND POSITIVE.
19475  107010  012701  000000          MOV     #000,R1         ;SET FD.
19476  107014  170101                  LDFPS   R1
19477
19478  107016  170720          TTB2:   NEGF    (R0)+           ;TEST INSTRUCTION.
19479
19480  107020  170205                  STFPS   R5              ;GET FPS.
19481  107022  012701  107062          MOV     #TTBTP1,R1              ;IS THE RESULT CORRECT.
19482  107026  012702  107072          MOV     #TTBTP2,R2
19483  107032  012703  000004          MOV     #4,R3
19484  107036  022122          1$:     CMP     (R1)+,(R2)+
19485  107040  001020                  BNE     TTB10           ;BRANCH IF INCORRECT.
19486  107042  077303                  SOB     R3,1$
19487  107044  022700  107066          CMP     #TTBTP1+4,R0    ;IS R0 CORRECT.
19488  107050  001014                  BNE     TTB10           ;BRANCH IF INCORRECT.
19489  107052  022705  000010          CMP     #010,R5         ;IS THE FPS CORRECT?
19490  107056  001011                  BNE     TTB10           ;BRANCH IF INCORRECT.
19491  107060  000411                  BR      TTBDONE
19492
19493                                  ;THESE ARE DATA TABLES AND A DATA BUFFER.
19494  107062  023245          TTBTP1: 023245
19495  107064  026720                  26720
19496  107066  122324                  122324
19497  107070  052672                  52672
19498  107072  123245          TTBTP2: 123245
19499  107074  026720                  26720
19500  107076  122324                  122324
19501  107100  052672                  52672
19502
19503  107102                  TTB10:
```

```
19504  107102  104000                          EMT                     ;
19505  107104                          TTBDONE:
19506  107104  004767  011070                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
19507                                                                   ;SEE IF THE USER HAS EXPRESSED
19508                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
19509                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19510                                                                   ;THE USER TYPED CONTROL G?).
19511                          ;*****************************************************************************
19512                          ;TEST 540       SPECIAL DEST, MODE2, GR7 (IMMEDIATE), TEST
19513                          ;*****************************************************************************
19514  107110                  TS540:
19515  107110  012700  107226          MOV     #UUBTP2,R0
19516  107114  012701  107154          MOV     #UUBTP1,R1      ;SET UP THE DATA BUFFER.
19517  107120  012702  000004          MOV     #4,R2
19518  107124  012021          1$:     MOV     (R0)+,(R1)+
19519  107126  077202                  SOB     R2,1$
19520  107130  012700  107154          MOV     #UUBTP1,R0
19521  107134  042737  100000  107154  BIC     #100000,@#UUBTP1        ;MAKE THE OPERAND POSITIVE.
19522  107142  012701  000200          MOV     #200,R1         ;SET FD.
19523  107146  170101                  LDFPS   R1
19524  107150  005001                  CLR     R1
19525
19526  107152  170727          UUB2:   NEGD    (R7)+           ;TEST INSTRUCTION.
19527  107154  005201  005201  005201  UUBTP1: 5201,5201,5201,5201
19528  107162  005201
19529                          ;NOTE THAT AFTER EXECUTING THIS INSTRUCTION R1 SHOULD CONTAIN 3.
19530  107164  170205                  STFPS   R5              ;GET FPS.
19531  107166  012703  107154          MOV     #UUBTP1,R3      ;IS THE RESULT CORRECT.
19532  107172  012702  107226          MOV     #UUBTP2,R2
19533  107176  012704  000004          MOV     #4,R4
19534  107202  022322          1$:     CMP     (R3)+,(R2)+
19535  107204  001014                  BNE     UUB10           ;BRANCH IF INCORRECT.
19536  107206  077403                  SOB     R4,1$
19537  107210  022701  000003          CMP     #3,R1           ;WAS R1 INCREMENTED CORRECTLY.
19538  107214  001010                  BNE     UUB10           ;BRANCH IF INCORRECT.
19539  107216  022705  000210          CMP     #210,R5         ;IS THE FPS CORRECT?
19540  107222  001005                  BNE     UUB10           ;BRANCH IF INCORRECT.
19541  107224  000405                  BR      UUBDONE
19542
19543                          ;THESE ARE DATA TABLE.
19544  107226  105201          UUBTP2: 105201
19545  107230  005201                  5201
19546  107232  005201                  5201
19547  107234  005201                  5201
19548
19549  107236                  UUB10:
19550  107236  104000                  EMT                     ;
19551  107240                  UUBDONE:
19552  107240  004767  010734          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
19553                                                          ;SEE IF THE USER HAS EXPRESSED
19554                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
19555                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19556                                                          ;THE USER TYPED CONTROL G?).
19557                          ;*****************************************************************************
19558                          ;TEST 541       SPECIAL DEST, MODE 6, TEST
19559                          ;*****************************************************************************
```

```
19560  107244
19561  107244  012701  107360          TS541:
19562  107250  012700  107370                  MOV     #XXBTP1,R1      ;SET UP THE DATA BUFFER.
19563  107254  012702  000004                  MOV     #XXBTP2,R0
19564  107260  012021                          MOV     #4,R2
19565  107262  077202                  1$:     MOV     (R0)+,(R1)+
19566  107264  012700  102157                  SOB     R2,1$
19567  107270  042737  100000  107360          MOV     #XXBTP1-5201,R0
19568  107276  012701  000200                  BIC     #100000,@#XXBTP1;MAKE OPERAND POSITIVE.
19569  107302  170101                          MOV     #200,R1         ;SET FD.
19570                                          LDFPS   R1
19571  107304  005001                          CLR     R1
19572  107306  170760  005201          XXB2:   NEGD    5201(R0)                ;TEST INSTRUCTION.
19573
19574  107312  170205                          STFPS   R5              ;GET FPS.
19575  107314  005701                          TST     R1
19576  107316  001030                          BNE     XXB10           ;WAS THE PC CORRECT AFTER EXECUTION?
19577  107320  012701  107360                  MOV     #XXBTP1,R1              ;IS THE RESULT CORRECT.
19578  107324  012702  107370                  MOV     #XXBTP2,R2
19579  107330  012703  000004                  MOV     #4,R3
19580  107334  022122                  1$:     CMP     (R1)+,(R2)+
19581  107336  001020                          BNE     XXB10           ;BRANCH IF INCORRECT.
19582  107340  077303                          SOB     R3,1$
19583  107342  022700  102157                  CMP     #XXBTP1-5201,R0 ;IS R0 CORRECT.
19584  107346  001014                          BNE     XXB10           ;BRANCH IF INCORRECT.
19585  107350  022705  000210                  CMP     #210,R5         ;IS THE FPS CORRECT?
19586  107354  001011                          BNE     XXB10           ;BRANCH IF INCORRECT.
19587  107356  000411                          BR      XXBDONE
19588
19589                                  ;THESE ARE DATA TABLES AND A DATA BUFFER.
19590  107360  023245                  XXBTP1: 023245
19591  107362  026720                          26720
19592  107364  122324                          122324
19593  107366  052672                          52672
19594  107370  123245                  XXBTP2: 123245
19595  107372  026720                          26720
19596  107374  122324                          122324
19597  107376  052672                          52672
19598
19599
19600  107400                          XXB10:
19601  107400  104000                          EMT                     ;
19602  107402                          XXBDONE:
19603  107402  004767  010572                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19604                                                                  ;SEE IF THE USER HAS EXPRESSED
19605                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19606                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19607                                                                  ;THE USER TYPED CONTROL G?).
19608
19609                                  ;*****************************************************************************
19610                                  ;TEST 542       SPECIAL DEST, MODE 7, TEST
19611                                  ;*****************************************************************************
19612  107406                          TS542:
19613
19614  107406  012701  107530                  MOV     #YYBTP1,R1      ;SET UP THE DATA BUFFER.
19615  107412  012700  107540                  MOV     #YYBTP2,R0
```

```
19616  107416  012702  000004          1$:     MOV     #4,R2
19617  107422  012021                          MOV     (R0)+,(R1)+
19618  107424  077202                          SOB     R2,1$
19619  107426  012700  102347                  MOV     #YYBTP3-5201,R0
19620  107432  012760  107530  005201          MOV     #YYBTP1,5201(R0)
19621  107440  042737  100000  107530          BIC     #100000,@#YYBTP1        ;MAKE THE OPERAND POSITIVE.
19622  107446  012701  000200                  MOV     #200,R1         ;SET FD.
19623  107452  170101                          LDFPS   R1
19624
19625  107454  005001                          CLR     R1
19626  107456  170770  005201          YYB2:   NEGD    @5201(R0)               ;TEST INSTRUCTION.
19627
19628  107462  170205                          STFPS   R5              ;GET FPS.
19629  107464  005701                          TST     R1              ;WAS THE PC CORRECT AFTER EXECUTION?
19630  107466  001031                          BNE     YYB10
19631  107470  012701  107530                  MOV     #YYBTP1,R1              ;IS THE RESULT CORRECT.
19632  107474  012702  107540                  MOV     #YYBTP2,R2
19633  107500  012703  000004                  MOV     #4,R3
19634  107504  022122          1$:     CMP     (R1)+,(R2)+
19635  107506  001021                          BNE     YYB10           ;BRANCH IF INCORRECT.
19636  107510  077303                          SOB     R3,1$
19637  107512  022700  102347                  CMP     #YYBTP3-5201,R0 ;IS R0 CORRECT.
19638  107516  001015                          BNE     YYB10           ;BRANCH IF INCORRECT.
19639  107520  022705  000210                  CMP     #210,R5         ;IS THE FPS CORRECT?
19640  107524  001012                          BNE     YYB10           ;BRANCH IF INCORRECT.
19641  107526  000412                          BR      YYBDONE
19642
19643                                  ;THESE ARE DATA TABLES AND A DATA BUFFER.
19644  107530  023245                  YYBTP1: 023245
19645  107532  026720                          26720
19646  107534  122324                          122324
19647  107536  052672                          52672
19648  107540  123245                  YYBTP2: 123245
19649  107542  026720                          26720
19650  107544  123324                          123324
19651  107546  052672                          52672
19652  107550  107530                  YYBTP3: YYBTP1
19653  107552                          YYB10:
19654  107552  104000                          EMT                     ;
19655
19656  107554                          YYBDONE:
19657  107554  004767  010420                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19658                                                                  ;SEE IF THE USER HAS EXPRESSED
19659                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19660                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19661                                                                  ;THE USER TYPED CONTROL G?).
19662                          ;*****************************************************************************
19663                          ;TEST 543       NEGD, ABSD AND TSTD TEST
19664                          ;*****************************************************************************
19665  107560                  TS543:
19666                          ;TEST NEGD WITH POS NONZERO OPERAND
19667  107560  004767  000526  WWB1:   JSR     PC,NATSUB
19668  107564  000000          1$:     0                               ;FLAG=NEGD.
19669  107566  016341          2$:     16341                           ;OPERAND.
19670  107570  055772                  55772
19671  107572  021133                  21133
```

```
19672  107574  055447                  55447
19673  107576  116341          3$:     116341                  ;RESULT.
19674  107600  055772                  55772
19675  107602  021133                  21133
19676  107604  055447                  55447
19677  107606  016341          4$:     16341                   ;ERROR RES.
19678  107610  055772                  55772
19679  107612  021133                  21133
19680  107614  055447                  55447
19681  107616  000207          5$:     207                     ;FPS BEFORE EXECUTION.
19682  107620  000210                  210                     ;FPS AFTER EXECUTION.
19683  107622  000200                  200                     ;ERROR FPS.
19684  107624  177777                  -1                      ;FEC
19685                          ;TEST NEGD WITH NEG OPERAND.
19686  107626  004767  000460  WWB2:   JSR     PC,NATSUB
19687  107632  000000          1$:     0                       ;FLAG=NEGD.
19688  107634  152525          2$:     152525                  ;OPERAND.
19689  107636  053545                  53545
19690  107640  055565                  55565
19691  107642  057505                  57505
19692  107644  052525          3$:     52525                   ;RESULT.
19693  107646  053545                  53545
19694  107650  055565                  55565
19695  107652  057505                  57505
19696  107654  152525          4$:     152525                  ;ERROR RES.
19697  107656  05354?                  53545
19698  107660  0555(                   55565
19699  107662  05750>                  57505
19700  107664  000217          5$:     217                     ;FPS BEFORE EXECUTION.
19701  107666  000200                  200                     ;FPS AFTER EXECUTION.
19702  107670  000210                  210                     ;ERROR FPS.
19703  107672  177777                  -1                      ;FEC
19704                          ;TEST ABSD WITH POSITIVE OPERAND
19705  107674  004767  000412  WWB3:   JSR     PC,NATSUB
19706  107700  000001          1$:     1                       ;FLAG=ABSD.
19707  107702  060705          2$:     60705                   ;OPERAND.
19708  107704  124735                  124735
19709  107706  060124                  60124
19710  107710  073560                  73560
19711  107712  060705          3$:     60705                   ;RESULT.
19712  107714  124735                  124735
19713  107716  060124                  60124
19714  107720  073560                  73560
19715  107722  160705          4$:     160705                  ;ERROR RES.
19716  107724  124735                  124735
19717  107726  060124                  60124
19718  107730  073560                  73560
19719  107732  000217          5$:     217                     ;FPS BEFORE EXECUTION.
19720  107734  000200                  200                     ;FPS AFTER EXECUTION.
19721  107736  000210                  210                     ;ERROR FPS.
19722  107740  177777                  -1                      ;EITHER BUT OP1B
19723                          ;TEST ABSD WITH NEG. OPERAND
19724  107742  004767  000344  WWB4:   JSR     PC,NATSUB
19725  107746  000001          1$:     1                       ;FLAG=ABSD.
19726  107750  154345          2$:     154345                  ;OPERAND.
19727  107752  076567                  76567
```

```
19728  107754  032123
19729  107756  043234
19730  107760  054345
19731  107762  076567
19732  107764  032123
19733  107766  043234
19734  107770  154345
19735  107772  076567
19736  107774  032123
19737  107776  043234
19738  110000  000217
19739  110002  000200
19740  110004  177777
19741  110006  177777
19742
19743  110010  004767  000276
19744  110014  000002
19745  110016  012321
19746  110020  045654
19747  110022  070107
19748  110024  034543
19749  110026  012321
19750  110030  045654
19751  110032  070107
19752  110034  034543
19753  110036  112321
19754  110040  045654
19755  110042  070107
19756  110044  034543
19757  110046  000217
19758  110050  000200
19759  110052  000210
19760  110054  177777
19761
19762  110056  004767  000230
19763  110062  000002
19764  110064  123765
19765  110066  023407
19766  110070  034510
19767  110072  045621
19768  110074  123765
19769  110076  023407
19770  110100  034510
19771  110102  045621
19772  110104  023765
19773  110106  023407
19774  110110  034510
19775  110112  045621
19776  110114  000207
19777  110116  000210
19778  110120  000200
19779  110122  177777
19780
19781  110124  004767  000162
19782  110130  000002
19783  110132  000175
```

```
        32123
        43234
3$:     54345                   ;RESULT.
        76567
        32123
        43234
4$:     154345                  ;ERROR RES.
        76567
        32123
        43234
5$:     217                     ;FPS BEFORE EXECUTION.
        200                     ;FPS AFTER EXECUTION.
        -1                      ;ERROR FPS.
        -1
;TEST WITH POSITIVE OP
WWB5:   JSR     PC,NATSUB
1$:     2                       ;FLAG=TSTD.
2$:     12321                   ;OPERAND.
        45654
        70107
        34543
3$:     12321                   ;RESULT.
        45654
        70107
        34543
4$:     112321                  ;ERROR RES.
        45654
        70107
        34543
5$:     217                     ;FPS BEFORE EXECUTION.
        200                     ;FPS AFTER EXECUTION.
        210                     ;ERROR FPS.
        -1
;TEST TSTD WITH NEG OP
WWB6:   JSR     PC,NATSUB
1$:     2                       ;FLAG=TSTD.
2$:     123765                  ;OPERAND.
        23407
        34510
        45621
3$:     123765                  ;RESULT.
        23407
        34510
        45621
4$:     23765                   ;ERROR RES.
        23407
        34510
        45621
5$:     207                     ;FPS BEFORE EXECUTION.
        210                     ;FPS AFTER EXECUTION.
        200                     ;ERROR FPS.
        -1
;TEST TSTD 0 OP
WWB7:   JSR     PC,NATSUB
1$:     2                       ;FLAG=TSTD.
2$:     175                     ;OPERAND.
```

```
19784  110134  176737                          176737
19785  110136  071727                          71727
19786  110140  037574                          37574
19787  110142  000175                  3$:     175                     ;RESULT.
19788  110144  176737                          176737
19789  110146  071727                          71727
19790  110150  037574                          37574
19791  110152  000000                  4$:     0                       ;ERROR RES.
19792  110154  000000                          0
19793  110156  000000                          0
19794  110160  000000                          0
19795  110162  000200                  5$:     200                     ;FPS BEFORE EXECUTION.
19796  110164  000204                          204                     ;FPS AFTER EXECUTION.
19797  110166  000214                          214                     ;ERROR FPS.
19798  110170  177777                          -1
19799                                  ;TEST TSTD -0 OP FIUV=0
19800  110172  004767  000114        WWB10:  JSR     PC,NATSUB
19801  110176  000002                  1$:     2                       ;FLAG=TSTD.
19802  110200  100123                  2$:     100123                  ;OPERAND.
19803  110202  021012                          21012
19804  110204  034565                          34565
19805  110206  043210                          43210
19806  110210  100123                  3$:     100123                  ;RESULT.
19807  110212  021012                          21012
19808  110214  034565                          34565
19809  110216  043210                          43210
19810  110220  000000                  4$:     0                       ;ERROR RES.
19811  110222  000000                          0
19812  110224  000000                          0
19813  110226  000000                          0
19814  110230  040203                  5$:     40203                   ;FPS BEFORE EXECUTION.
19815  110232  040214                          040214                  ;FPS AFTER EXECUTION.
19816  110234  140214                          140214                  ;ERROR FPS.
19817  110236  177777                          -1
19818                                  ;TEST TSTD -0 OP FIUV=1
19819  110240  004767  000046        WWB11:  JSR     PC,NATSUB
19820  110244  000002                  1$:     2                       ;FLAG=TSTD.
19821  110246  100137                  2$:     100137                  ;OPERAND.
19822  110250  024613                          24613
19823  110252  057024                          57024
19824  110254  060137                          60137
19825  110256  100137                  3$:     100137                  ;RESULT.
19826  110260  024613                          24613
19827  110262  057024                          57024
19828  110264  060137                          60137
19829  110266  000000                  4$:     0                       ;ERROR RES.
19830  110270  000000                          0
19831  110272  000000                          0
19832  110274  000000                          0
19833  110276  044200                  5$:     44200                   ;FPS BEFORE EXECUTION.
19834  110300  144214                          144214                  ;FPS AFTER EXECUTION.
19835  110302  044214                          044214                  ;ERROR FPS.
19836  110304  000014                          14
19837  110306  000167  000162                  JMP     WWBDONE
19838
19839                                  ;THIS SUBROUTINE, NATSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
```

```
                        ;THE EITHER A TSTD, AN ABSD OR A NEGD INSTRUCTION AND CHECK THE RESULTS. A CALL
                        ;TO IT IS MADE THUS:
                        ;
                        ;               JSR     PC,@#NATSUB
                        ;               FLAG:   .WORD   X               ;INSTRUCTION TYPE FLAG.
                        ;               ACARG:  .WORD   X,X,X,X         ;OPERAND
                        ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
                        ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
                        ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
                        ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
                        ;               FEC:    .WORD   X               ;EXPECTED FEC
                        ;               ERFPS:  .WORD   X               ;ERROR FPS.
                        ;               ERR1:   ERROR   X               ;DATA ERROR.
                        ;                       BR      CONT
                        ;               ERR2:   ERROR   X               ;FPS ERROR.
                        ;               CONT:                           ;RETURN ADDRESS
                        ;
                        ;THE OPERAND IS SET UP IN NATBF1. THEN
                        ;THE EITHER THE TSTD, NEGD OR ABSD INSTRUCTION IS EXECUTED.
                        ;NATSUB USES THE FIRST OPERAND AS A FLAG TO DETERMINE WHICH INSTRUCTION
                        ;IS TO BE EXECUTED: 0 = NEGD, 1 = ABSD, 2 = TSTD.
                        ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                        ;COMPARED WITH FPSA. IF THIS TOO IS CORRECT NATSUB RETURNS CONTROL
                        ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD NATSUB
                        ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN NATSUB WILL RETURN
                        ;TO THE ERROR CALL AT ERR2, OTHERWISE NATSUB ITSELF
                        ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                        ;INSTRUCTION IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                        ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                        ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN NATSUB
                        ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                        ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND NATSUB WILL
                        ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.


110312  012601          NATSUB: MOV     (SP)+,R1                ;GET A POINTER TO THE ARGUMENTS.
110314  010102                  MOV     R1,R2           ;COPY THE OPERAND.
110316  062702  000002          ADD     #2,R2
110322  012703  110462          MOV     #NATBF1,R3
110326  012704  000004          MOV     #4,R4
110332  012223          1$:     MOV     (R2)+,(R3)+
110334  077402                  SOB     R4,1$
110336  016100  000032          MOV     32(R1),R0       ;LOAD THE FPS.
110342  170100                  LDFPS   R0
110344  012700  110462          MOV     #NATBF1,R0      ;SET UP THE OPERAND ADDRESS.
110350  011102                  MOV     (R1),R2         ;GET THE FLAG TO DETERMINE WHICH
110352  006302                  ASL     R2              ;INSTRUCTION TO EXECUTE.
110354  006302                  ASL     R2              ;0 = NEGD, 1 = ABSD, 2 = TSTD
110356  012703  110366          MOV     #NATINS,R3
110362  060203                  ADD     R2,R3
110364  000113                  JMP     (R3)            ;GO EXECUTE THE INSTRUCTION.
110366  170710          NATINS: NEGD    (R0)
110370  000403                  BR      2$
110372  170610                  ABSD    (R0)
110374  000401                  BR      2$
110376  170510                  TSTD    (R0)
```

```
19896
19897  110400  170204                  2$:     STFPS   R4              ;GET THE FPS.
19898  110402  170305                          STST    R5              ;GET THE FEC.
19899  110404  010100                          MOV     R1,R0           ;WAS THE RESULT CORRECT?
19900  110406  062700  000012                ADD     #12,R0
19901  110412  012702  110462                MOV     #NATBF1,R2
19902  110416  012703  000004                MOV     #4,R3
19903  110422  022022                  3$:     CMP     (R0)+,(R2)+
19904  110424  001014                          BNE     10$             ;BRANCH IF INCORRECT.
19905  110426  077303                          SOB     R3,3$
19906  110430  026104  000034                CMP     34(R1),R4       ;WAS THE FPS CORRECT?
19907  110434  001010                          BNE     10$             ;BRANCH IF INCORRECT.
19908  110436  005761  000034                TST     34(R1)          ;IF THE EXPECTED FPS WAS NEGATIVE CHECK THE FEC.
19909  110442  100003                          BPL     4$
19910  110444  026105  000040                CMP     40(R1),R5       ;WAS THE FEC CORRECT.
19911  110450  001002                          BNE     10$             ;BRANCH IF INCORRECT.
19912  110452  000161  000042          4$:     JMP     42(R1)          ;RETURN.
19913
19914  110456                          10$:
19915  110456  104000                          EMT                     ;
19916
19917  110460  177777                          .WORD   -1
19918  110462  177777  177777  177777 NATBF1: .WORD   -1,-1,-1,-1,-1
19919  110470  177777  177777
19920
19921  110474                          WWBDONE:
19922  110474  004767  007500                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
19923                                                                  ;SEE IF THE USER HAS EXPRESSED
19924                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19925                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19926                                                                  ;THE USER TYPED CONTROL G?).
19927
19928
19929
19930                                  ;*****************************************************************************************
19931                                  ;TEST 544       SOURCE MODES, MODE 1 (FL=0), TEST
19932                                  ;*****************************************************************************************
19933  110500                          TS544:
19934
19935
19936
19937  110500  012700  110550                  MOV     #AACTP1,R0      ;SET UP TEST DATA IN BUFFER.
19938  110504  012710  147517                  MOV     #147517,(R0)
19939  110510  012737  110524  037244          MOV     #AAC2,@#$TMP2
19940  110516  012737  110554  000004          MOV     #AAC10,@#ERRVECT        ;SET UP FOR TRAPS TO 4.
19941  110524  170110                  AAC2:   LDFPS   (R0)            ;TEST INSTRUCTION.
19942
19943  110526  170205                          STFPS   R5              ;GET FPS
19944
19945  110530  020027  110550                  CMP     R0,#AACTP1      ;IS R0 CORRECT?
19946  110534  001007                          BNE     AAC10           ;BR IF NOT.
19947  110536  022705  147517                  CMP     #147517,R5      ;IS FPS CORRECT?
19948  110542  001004                          BNE     AAC10           ;BR IF NOT.
19949  110544  000404                          BR      AACDONE
19950
19951                                  ;TEST BUFFER AND DATA:
```

```
19952  110546  177777                          -1
19953  110550  147517                AACTP1: 147517
19954  110552  177777                          -1
19955  110554                        AAC10:
19956  110554  104000                          EMT                     ;
19957
19958  110556                        AACDONE:
19959  110556  004767  007416                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
19960                                                                  ;SEE IF THE USER HAS EXPRESSED
19961                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19962                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19963                                                                  ;THE USER TYPED CONTROL G?).
19964
19965
19966                                 ;*****************************************************************************
19967                                 ;TEST 545       SOURCE MODES, MODE 2 (FL=0), TEST
19968                                 ;*****************************************************************************
19969  110562                        TS545:
19970
19971
19972  110562  012700  110624                  MOV     #BBCTP1,R0      ;SET UP TEST DATA IN BUFFER.
19973  110566  012710  145212                  MOV     #145212,(R0)
19974  110572  012737  110630  000004          MOV     #BBC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
19975
19976  110600  170120                BBC2:   LDFPS   (R0)+           ;TEST INSTRUCTION.
19977
19978  110602  170205                          STFPS   R5              ;GET FPS
19979
19980  110604  020027  110626                  CMP     R0,#BBCTP1+2    ;IS R0 CORRECT?
19981  110610  001007                          BNE     BBC10           ;BR IF NOT.
19982  110612  022705  145212                  CMP     #145212,R5      ;IS THE FPS CORRECT?
19983  110616  001004                          BNE     BBC10           ;BR IF NOT.
19984  110620  000404                          BR      BBCDONE
19985
19986
19987                                 ;TEST BUFFER AND DATA:
19988  110622  177777                          -1
19989  110624  177777                BBCTP1: .WORD   -1
19990  110626  177777                          -1
19991
```

```
19992
19993  110630
19994  110630  104000                  BBC10:
                                               EMT                     ;
19995  110632                          BBCDONE:
19996  110632  004767  007342                 JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
                                                                       ;SEE IF THE USER HAS EXPRESSED
19997                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
19998                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
19999                                                                  ;THE USER TYPED CONTROL G?).
20000
20001
20002
20003                                  ;*********************************************************************************
20004                                  ;TEST 546       SOURCE MODES, MODE 4 (FL=0), TEST
20005                                  ;*********************************************************************************
20006  110636                          TS546:
20007
20008
20009  110636  012700  110720                 MOV     #DDCTP1+2,R0    ;SET UP THE TEST DATA BUFFER.
20010  110642  012760  105252  177776         MOV     #105252,-2(R0)
20011  110650  012737  110664  037244         MOV     #DDC2,@#$TMP2
20012  110656  012737  110730  000004         MOV     #DDC10,@#ERRVEC
20013  110664  170140                  DDC2:  LDFPS   -(R0)
20014  110666  170205                         STFPS   R5
20015  110670  020027  110716                 CMP     R0,#DDCTP1
20016  110674  001015                         BNE     DDC10
20017  110676  022705  105252                 CMP     #105252,R5
20018  110702  001012                         BNE     DDC10
20019  110704  000412                         BR      DDCDONE
20020
20021  110706  177777  177777  177777         -1,-1,-1,-1
20022  110714  177777
20023  110716  177777                  DDCTP1: -1
20024  110720  177777  177777  177777         -1,-1,-1,-1
20025  110726  177777
20026  110730                          DDC10:
20027  110730  104000                         EMT                     ;
20028  110732                          DDCDONE:
20029  110732  004767  007242                 JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
20030                                                                  ;SEE IF THE USER HAS EXPRESSED
20031                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
20032                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20033                                                                  ;THE USER TYPED CONTROL G?).
20034                                  ;*********************************************************************************
20035                                  ;TEST 547       SOURCE MODES, MODE 3 (FL=0), TEST
20036                                  ;*********************************************************************************
20037  110736                          TS547:
20038  110736  012700  111024                 MOV     #EECTP2,R0
20039  110742  012710  111014                 MOV     #EECTP1,(R0)
20040  110746  012767  103456  000040         MOV     #103456,EECTP1
20041  110754  012737  111036  000004         MOV     #EEC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
20042  110762  170130                  EEC2:  LDFPS   @(R0)+          ;TEST INSTRUCTION.
20043  110764  170205                         STFPS   R5              ;GET THE FPS.
20044  110766  020027  111026                 CMP     R0,#EECTP2+2    ;IS R0 CORRECT?
20045  110772  001021                         BNE     EEC10           ;BR IF NOT.
20046  110774  022705  103456                 CMP     #103456,R5      ;IS THE FPS CORRECT?
20047  111000  001016                         BNE     EEC10           ;BR IF NOT.
```

```
20048  111002  000416                          BR      EECDONE
20049
20050
20051                                          ;TEST BUFFER AND DATA:
20052  111004  177777  177777  177777                  -1,-1,-1,-1
20053  111012  177777
20054  111014  177777                          EECTP1: -1
20055  111016  177777  177777  177777                  -1,-1,-1
20056  111024  111014  177777  177777          EECTP2: EECTP1,-1,-1,-1,
20057  111032  177777  000000
20058
20059  111036                                  EEC10:
20060  111036  104000                                  EMT                     ;
20061  111040                                  EECDONE:
20062  111040  004767  007134                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20063                                                                          ;SEE IF THE USER HAS EXPRESSED
20064                                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
20065                                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20066                                                                          ;THE USER TYPED CONTROL G?).
20067                                          ;*****************************************************************************
20068                                          ;TEST 550       SOURCE MODES, MODE 5 (FL=0), TEST
20069                                          ;*****************************************************************************
20070  111044                                  TS550:
20071  111044  012700  111130                          MOV     #FFCTP2+2,R0    ;SET UP THE TEST DATA BUFFER.
20072  111050  012760  111116  177776                  MOV     #FFCTP1,-2(R0)
20073  111056  012737  045412  111116                  MOV     #45412,@#FFCTP1
20074  111064  012737  111136  000004                  MOV     #FFC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
20075  111072  170150                          FFC2:   LDFPS   @-(R0)          ;TEST INSTRUCTION.
20076  111074  170205                                  STFPS   R5              ;GET THE FPS.
20077  111076  020027  111126                          CMP     R0,#FFCTP2      ;IS R0 CORRECT?
20078  111102  001015                                  BNE     FFC10           ;BR IF NOT.
20079  111104  022705  045412                          CMP     #45412,R5       ;IS THE FPS CORRECT?
20080  111110  001012                                  BNE     FFC10           ;BR IF NOT.
20081  111112  000412                                  BR      FFCDONE
20082
20083
20084                                          ;TEST BUFFER AND DATA:
20085  111114  177777                                  -1
20086  111116  177777                          FFCTP1: -1
20087  111120  177777  177777  177777                  -1,-1,-1
20088  111126  111116  177777  177777          FFCTP2: FFCTP1,-1,-1,-1
20089  111134  177777
20090
20091  111136                                  FFC10:
20092  111136  104000                                  EMT                     ;
20093  111140                                  FFCDONE:
20094  111140  004767  007034                          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20095                                                                          ;SEE IF THE USER HAS EXPRESSED
20096                                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
20097                                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20098                                                                          ;THE USER TYPED CONTROL G?).
20099                                          ;*****************************************************************************
20100                                          ;TEST 551       SOURCE MODES, MODE 6 (FL=0), TEST
20101                                          ;*****************************************************************************
20102  111144                                  TS551:
20103  111144  012700  104017                          MOV     #GGCTP1-5201,R0 ;SET UP THE TEST DATA BUFFER.
```

```
20104  111150  012737  046543  111220          MOV     #46543,@#GGCTP1
20105  111156  005001                          CLR     R1
20106  111160  012737  111232  000004          MOV     #GGC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
20107  111166  170160  005201          GGC2:   LDFPS   5201(R0)                 ;TEST INSTRUCTION.
20108  111172  170204                          STFPS   R4              ;GET THE FPS.
20109  111174  005701                          TST     R1              ;WAS PC CORRECT AFTER EXECUTION?
20110  111176  001015                          BNE     GGC10           ;BR IF NOT.
20111  111200  020027  104017                  CMP     R0,#GGCTP1-5201 ;IS R0 CORRECT?
20112  111204  001012                          BNE     GGC10           ;BR IF NOT.
20113  111206  022704  046543                  CMP     #46543,R4       ;IS THE FPS CORRECT?
20114  111212  001007                          BNE     GGC10           ;BR IF NOT.
20115  111214  000407                          BR      GGCDONE
20116
20117
20118                                  ;TEST BUFFER AND DATA:
20119  111216  177777                          -1
20120  111220  177777  177777  177777  GGCTP1: -1,-1,-1,-1
20121  111226  177777
20122  111230  177777                          -1
20123  111232                          GGC10:
20124  111232  104000                          EMT                     ;
20125  111234                          GGCDONE:
20126  111234  004767  006740                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20127                                                                  ;SEE IF THE USER HAS EXPRESSED
20128                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
20129                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20130                                                                  ;THE USER TYPED CONTROL G?).
20131                                  ;*****************************************************************************
20132                                  ;TEST 552       SOURCE MODES, MODE 7 (FL=0), TEST
20133                                  ;*****************************************************************************
20134  111240                          TS552:
20135  111240  012700  104131                  MOV     #HHCTP2-5201,R0 ;SET UP THE TEST DATA BUFFER.
20136  111244  012760  111322  005201          MOV     #HHCTP1,5201(R0)
20137  111252  012737  004547  111322          MOV     #4547,@#HHCTP1
20138  111260  005001                          CLR     R1
20139  111262  012737  111342  000004          MOV     #HHC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
20140  111270  170170  005201          HHC2:   LDFPS   @5201(R0)                ;TEST INSTRUCTION.
20141  111274  170204                          STFPS   R4              ;GET THE FPS.
20142  111276  005701                          TST     R1              ;WAS PC CORRECT AFTER EXECUTION?
20143  111300  001020                          BNE     HHC10           ;BR IF NOT.
20144  111302  020027  104131                  CMP     R0,#HHCTP2-5201 ;IS R0 CORRECT?
20145  111306  001015                          BNE     HHC10           ;BR IF NOT.
20146  111310  022704  004547                  CMP     #4547,R4        ;IS THE FPS CORRECT?
20147  111314  001012                          BNE     HHC10           ;BR IF NOT.
20148  111316  000412                          BR      HHCDONE
20149
20150
20151                                  ;TEST BUFFER AND DATA:
20152  111320  177777                          -1
20153  111322  177777  177777  177777  HHCTP1: .WORD -1,-1,-1,-1
20154  111330  177777
20155  111332  177777  177777  177777  HHCTP2: .WORD -1,-1,-1,-1
20156  111340  177777
20157  111342                          HHC10:
20158  111342  104000                          EMT                     ;
20159  111344                          HHCDONE:
```

```
20160  111344  004767  006630                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20161                                                                   ;SEE IF THE USER HAS EXPRESSED
20162                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
20163                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20164                                                                   ;THE USER TYPED CONTROL G?).
20165
20166
20167
20168                                   ;*****************************************************************************
20169                                   ;TEST 553       SOURCE MODES, MODE 2 GR7 (FL=1), TEST
20170                                   ;*****************************************************************************
20171  111350                           TS553:
20172
20173  111350  012737  111406  000004           MOV     #IIC20,@#ERRVECT ;SET UP FOR TRAPS TO 4.
20174  111356  012700  000300                   MOV     #300,R0
20175  111362  170100                           LDFPS   R0
20176  111364  005001                           CLR     R1
20177
20178  111366  177027                   IIC2:   LDCLD   (R7)+,AC0               ;TEST INSTRUCTION.
20179  111370  005201                           5201
20180  111372  005201                           5201
20181  111374  005201                           5201
20182  111376  005201                           5201
20183
20184  111400  020127  000003                   CMP     R1,#3           ;WAS PC CORRECT AFTER EXECUTION?
20185  111404  001401                           BEQ     IICDONE
20186  111406                           IIC20:
20187  111406  104000                           EMT                     ;
20188
20189  111410                           IICDONE:
20190  111410  004767  006564                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20191                                                                   ;SEE IF THE USER HAS EXPRESSED
20192                                                                   ;THE DESIRE TO CHANGE THE SOFTWARE
20193                                                                   ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20194                                                                   ;THE USER TYPED CONTROL G?).
20195
20196
20197                                   ;*****************************************************************************
20198                                   ;TEST 554       SOURCE MODES, MODE 2 (FL=1), TEST
20199                                   ;*****************************************************************************
20200  111414                           TS554:
20201
20202  111414  012700  000300                   MOV     #300,R0
20203  111420  170100                           LDFPS   R0
20204  111422  012700  111466                   MOV     #TCCBF0,R0      ;SET UP THE TEST DATA BUFFER.
20205  111426  177020                   TCC2:   LDCLD   (R0)+,AC0               ;TEST INSTRUCTION.
20206
20207  111430  170204                           STFPS   R4              ;GET THE FPS.
20208  111432  012701  111476                   MOV     #TCCBF1,R1      ;GET THE RESULT.
20209  111436  012702  000200                   MOV     #200,R2
20210  111442  170102                           LDFPS   R2
20211  111444  174011                           STD     AC0,(R1)
20212  111446  020027  111472                   CMP     R0,#TCCBF0+4    ;IS R0 CORRECT?
20213  111452  001401                           BEQ     TCC3
20214  111454  104000                           EMT                     ;
20215
```

```
20216  111456  022704  000300          TCC3:   CMP     #300,R4 ;IS THE FPS CORRECT?
20217  111462  001411                          BEQ     TCCDONE
20218  111464  104000                          EMT                     ;
20219
20220
20221                                  ;TEST BUFFER AND DATA:
20222  111466  001234  067076  054321  TCCBF0: .WORD   01234,67076,54321,012345
20223  111474  012345
20224  111476  177777  177777  177777  TCCBF1: -1,-1,-1,-1
20225  111504  177777
20226
20227  111506                          TCCDONE:
20228  111506  004767  006466                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20229                                                                  ;SEE IF THE USER HAS EXPRESSED
20230                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
20231                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20232                                                                  ;THE USER TYPED CONTROL G?).
20233
20234
20235
20236                                  ;*****************************************************************************
20237                                  ;TEST 555       LDCIF AND LDCLF TEST
20238                                  ;*****************************************************************************
20239  111512                          TS555:
20240
20241
20242                                  ;ZERO   OPERAND FL=0
20243
20244  111512  004737  112404          KKC1:   JSR     PC,@#LDCFSUB    ;GO EXECUTE INSTRUCTION.
20245
20246  111516  000000  000000          1$:     .WORD   0,0             ;FSRC OPERAND.
20247  111522  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
20248  111526  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
20249  111532  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
20250  111534  000004                          4                       ;FPS AFTER EXECUTION.
20251  111536  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
20252                                  ;ZERO   OPERAND FL=0
20253
20254  111540  004737  112404          KKC2:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20255
20256  111544  000000  177777          1$:     .WORD   0,-1            ;FSRC OPERAND.
20257  111550  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
20258  111554  004177  177400          3$:     4177,177400             ;ANTICIPATED ERRONEOUS RESULT.
20259  111560  000000                  4$:     0                       ;FPS BEFORE EXECUTION.
20260  111562  000004                          4                       ;FPS AFTER EXECUTION.
20261  111564  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
20262                                  ;ZERO   OPERAND FL=1
20263
20264  111566  004737  112404          KKC3:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20265
20266  111572  000000  000000          1$:     .WORD   0,0             ;FSRC OPERAND.
20267  111576  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
20268  111602  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
20269  111606  000100                  4$:     100                     ;FPS BEFORE EXECUTION.
20270  111610  000104                          104                     ;FPS AFTER EXECUTION.
20271  111612  000004                          4                       ;ANTICIPATED ERRONEOUS FPS.
```

```
20272
20273  111614  004737  112404  KKC4:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20274  111620  040000  000000  1$:     .WORD   40000,0         ;FSRC OPERAND.
20275  111624  043600  000000  2$:     .WORD   43600,0         ;EXPECTED RESULT.
20276  111630  047600  000000  3$:     .WORD   47600,0         ;ANTICIPATED ERRONEOUS RESULT.
20277  111634  000017          4$:     17                      ;FPS BEFORE EXECUTION.
20278  111636  000000                  0                       ;FPS AFTER EXECUTION.
20279  111640  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
20280
20281  111642  004737  112404  KKC5:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20282  111646  000001  000000  1$:     .WORD   1,0             ;FSRC OPERAND.
20283  111652  040200  000000  2$:     .WORD   40200,0         ;EXPECTED RESULT.
20284  111656  044200  000000  3$:     .WORD   44200,0         ;ANTICIPATED ERRONEOUS RESULT.
20285  111662  000017          4$:     17                      ;FPS BEFORE EXECUTION.
20286  111664  000000                  0                       ;FPS AFTER EXECUTION.
20287  111666  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
20288
20289  111670  004737  112404  KKC6:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20290  111674  000252  000000  1$:     .WORD   252,0           ;FSRC OPERAND.
20291  111700  042052  000000  2$:     .WORD   42052,0         ;EXPECTED RESULT.
20292  111704  046052  000000  3$:     .WORD   46052,0         ;ANTICIPATED ERRONEOUS RESULT.
20293  111710  000000          4$:     0                       ;FPS BEFORE EXECUTION.
20294  111712  000000                  0                       ;FPS AFTER EXECUTION.
20295  111714  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
20296
20297  111716  004737  112404  KKC7:   JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20298  111722  140000  000000  1$:     .WORD   -40000,0        ;FSRC OPERAND.
20299  111726  143600  000000  2$:     .WORD   143600,0        ;EXPECTED RESULT.
20300  111732  043600  000000  3$:     .WORD   43600,0         ;ANTICIPATED ERRONEOUS RESULT.
20301  111736  000007          4$:     7                       ;FPS BEFORE EXECUTION.
20302  111740  000010                  10                      ;FPS AFTER EXECUTION.
20303  111742  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
20304
20305  111744  004737  112404  KKC10:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20306  111750  177777  000000  1$:     .WORD   -1,0            ;FSRC OPERAND.
20307  111754  140200  000000  2$:     .WORD   140200,0        ;EXPECTED RESULT.
20308  111760  144000  000400  3$:     .WORD   144000,400      ;ANTICIPATED ERRONEOUS RESULT.
20309  111764  000000          4$:     0                       ;FPS BEFORE EXECUTION.
20310  111766  000010                  10                      ;FPS AFTER EXECUTION.
20311  111770  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
20312
20313  111772  004737  112404  KKC11:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20314  111776  125252  000000  1$:     .WORD   125252,0        ;FSRC OPERAND.
20315  112002  143652  126000  2$:     .WORD   143652,126000   ;EXPECTED RESULT.
20316  112006  043652  126000  3$:     .WORD   43652,126000    ;ANTICIPATED ERRONEOUS RESULT.
20317  112012  000007          4$:     7                       ;FPS BEFORE EXECUTION.
20318  112014  000010                  10                      ;FPS AFTER EXECUTION.
20319  112016  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
20320
20321  112020  004737  112404  KKC12:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20322  112024  040000  000000  1$:     .WORD   40000,0         ;FSRC OPERAND.
20323  112030  047600  000000  2$:     .WORD   47600,0         ;EXPECTED RESULT.
20324  112034  043600  000000  3$:     .WORD   43600,0         ;ANTICIPATED ERRONEOUS RESULT.
20325  112040  000117          4$:     117                     ;FPS BEFORE EXECUTION.
20326  112042  000100                  100                     ;FPS AFTER EXECUTION.
20327  112044  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
```

;OPERAND POSITIVE FL=0

;OPERAND=1, FL=0

;OPERAND= PATTERN FL=0

;OPERAND=-40000 FL=0

;OPERAND=-1 FL-0

;OPERAND=PATTERN FL=0

;OPERAND POS FL-1

```
20328
20329  112046  004737  112404  ;OPERAND=1          FL=1
20330  112052  000000  000001  KKC13:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20331  112056  040200  000000  1$:     .WORD   0,1             ;FSRC OPERAND.
20332  112062  034200  000000  2$:     .WORD   40200,0         ;EXPECTED RESULT.
20333  112066  000100          3$:     .WORD   34200,0         ;ANTICIPATED ERRONEOUS RESULT.
20334  112070  000100          4$:     100                     ;FPS BEFORE EXECUTION.
20335  112072  177777                  100                     ;FPS AFTER EXECUTION.
20336                                   -1                      ;ANTICIPATED ERRONEOUS FPS.
20337  112074  004737  112404  ;OPERAND=           PATTERN FL=1
20338  112100  000000  000252  KKC14:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20339  112104  042052  000000  1$:     .WORD   0,252           ;FSRC OPERAND.
20340  112110  036052  000000  2$:     .WORD   42052,0         ;EXPECTED RESULT.
20341  112114  000111          3$:     .WORD   36052,0         ;ANTICIPATED ERRONEOUS RESULT.
20342  112116  000100          4$:     111                     ;FPS BEFORE EXECUTION.
20343  112120  177777                  100                     ;FPS AFTER EXECUTION.
20344                                   -1                      ;ANTICIPATED ERRONEOUS FPS.
20345  112122  004737  112404  ;OPERAND=-40000,0       FL=1
20346  112126  140000  000000  KKC15:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20347  112132  147600  000000  1$:     .WORD   -40000,0                ;FSRC OPERAND.
20348  112136  047600  000000  2$:     .WORD   147600,0                ;EXPECTED RESULT.
20349  112142  000107          3$:     .WORD   47600,0         ;ANTICIPATED ERRONEOUS RESULT.
20350  112144  000110          4$:     107                     ;FPS BEFORE EXECUTION.
20351  112146  177777                  110                     ;FPS AFTER EXECUTION.
20352                                   -1                      ;ANTICIPATED ERRONEOUS FPS.
20353  112150  004737  112404  ;OPERAND=-1,-1   FL=1
20354  112154  177777  177777  KKC16:  JSR     PC,@#LDCFSUB    ;GO EXECUTE THE INSTRUCTION.
20355  112160  140200  000000  1$:     .WORD   -1,-1           ;FSRC OPERAND.
20356  112164  150000  000000  2$:     .WORD   140200,0                ;EXPECTED RESULT.
20357  112170  000100          3$:     .WORD   150000,0                ;ANTICIPATED ERRONEOUS RESULT.
20358  112172  000110          4$:     100                     ;FPS BEFORE EXECUTION.
20359  112174  177777                  110                     ;FPS AFTER EXECUTION.
20360                                   -1                      ;ANTICIPATED ERRONEOUS FPS.
20361  112176  004737  112404  ;OPERAND=-PATTERN        FL=1,
20362  112202  125252  125252  KKC17:  JSR     PC,@#LDCFSUB    ROUND MODE
20363  112206  147652  125253  1$:     .WORD   125252,125252   ;GO EXECUTE THE INSTRUCTION.
20364  112212  047652  125253  2$:     .WORD   147652,125253           ;FSRC OPERAND.
20365  112216  000105          3$:     .WORD   47652,125253            ;EXPECTED RESULT.
20366  112220  000110          4$:     105                     ;ANTICIPATED ERRONEOUS RESULT.
20367  112222  177777                  110                     ;FPS BEFORE EXECUTION.
20368                                   -1                      ;FPS AFTER EXECUTION.
20369  112224  004737  112404  ;OPERAND=77777,177500    FL=1,  ;ANTICIPATED ERRONEOUS FPS.
20370  112230  077777  177500  KKC20:  JSR     PC,@#LDCFSUB    ROUND MODE
20371  112234  047777  177777  1$:     .WORD   77777,177500    ;GO EXECUTE THE INSTRUCTION.
20372  112240  047777  177776  2$:     .WORD   47777,177777            ;FSRC OPERAND.
20373  112244  000117          3$:     .WORD   47777,177776            ;EXPECTED RESULT.
20374  112246  000100          4$:     117                     ;ANTICIPATED ERRONEOUS RESULT.
20375  112250  177777                  100                     ;FPS BEFORE EXECUTION.
20376                                   -1                      ;FPS AFTER EXECUTION.
20377  112252  004737  112404  ;OPERAND=40000,000100    FL=1,  ;ANTICIPATED ERRONEOUS FPS.
20378  112256  040000  000100  KKC21:  JSR     PC,@#LDCFSUB    ROUND MODE
20379  112262  047600  000001  1$:     .WORD   40000,100       ;GO EXECUTE THE INSTRUCTION.
20380  112266  047600  000000  2$:     .WORD   47600,1                 ;FSRC OPERAND.
20381  112272  000102          3$:     .WORD   47600,0         ;EXPECTED RESULT.
20382  112274  000100          4$:     102                     ;ANTICIPATED ERRONEOUS RESULT.
20383  112276  177777                  100                     ;FPS BEFORE EXECUTION.
                                        -1                      ;FPS AFTER EXECUTION.
                                                                ;ANTICIPATED ERRONEOUS FPS.
```

```
20384
20385  112300  004737  112404  ;OPERAND=40000,000100   FL=1,   TRUNC MODE
20386  112304  040000  000100  KKC22:  JSR     PC,@#LDCFSUB   ;GO EXECUTE THE INSTRUCTION.
20387  112310  047600  000000  1$:     .WORD   40000,100      ;FSRC OPERAND.
20388  112314  047600  000001  2$:     .WORD   47600,0        ;EXPECTED RESULT.
20389  112320  000157          3$:     .WORD   47600,1        ;ANTICIPATED ERRONEOUS RESULT.
20390  112322  000140          4$:     157                    ;FPS BEFORE EXECUTION.
20391  112324  177777                  140                    ;FPS AFTER EXECUTION.
20392                                   -1                     ;ANTICIPATED ERRONEOUS FPS.
20393  112326  004737  112404  ;OPERAND=100000,0 (MOST NEG #)  FL=0
20394  112332  100000  000000  KKC23:  JSR     PC,@#LDCFSUB   ;GO EXECUTE THE INSTRUCTION.
20395  112336  144000  000000  1$:     .WORD   100000,0       ;FSRC OPERAND.
20396  112342  143600  000000  2$:     .WORD   144000,0       ;EXPECTED RESULT.
20397  112346  000007          3$:     .WORD   143600,0       ;ANTICIPATED ERRONEOUS RESULT.
20398  112350  000010          4$:     7                      ;FPS BEFORE EXECUTION.
20399  112352  177777                  10                     ;FPS AFTER EXECUTION.
20400                                   -1                     ;ANTICIPATED ERRONEOUS FPS.
20401  112354  004737  112404  ;OPERAND=100000,0       FL=1
20402  112360  100000  000000  KKC24:  JSR     PC,@#LDCFSUB   ;GO EXECUTE THE INSTRUCTION.
20403  112364  150000  000000  1$:     .WORD   100000,0       ;FSRC OPERAND.
20404  112370  147600  000000  2$:     .WORD   150000,0       ;EXPECTED RESULT.
20405  112374  000107          3$:     .WORD   147600,0       ;ANTICIPATED ERRONEOUS RESULT.
20406  112376  000110          4$:     107                    ;FPS BEFORE EXECUTION.
20407  112400  177777                  110                    ;FPS AFTER EXECUTION.
20408  112402  000441                  -1                     ;ANTICIPATED ERRONEOUS FPS.
20409                          6$:     BR      KKCDONE
20410
20411  ;THIS SUBROUTINE, LDCFSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
20412  ;THE LDCIF OR LDCLF INSTRUCTION AND CHECK THE RESULTS. A CALL
20413  ;TO IT IS MADE THUS:
20414  ;
20415  ;               JSR     PC,@#LDCFSUB
20416  ;               ACARG:  .WORD   X,X             ;AC OPERAND
20417  ;               RES:    .WORD   X,X             ;EXPECTED RESULT
20418  ;               ERRES:  .WORD   X,X             ;ERROR RESULT
20419  ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
20420  ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
20421  ;               ERFPS:  .WORD   X               ;ERROR FPS
20422  ;               ERR1:   ERROR   X               ;DATA ERROR
20423  ;                       BR      CONT
20424  ;               ERR2:   ERROR   X               ;FPS ERROR
20425  ;               CONT:                           ;RETURN ADDRESS
20426  ;
20427  ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
20428  ;THE LDCIF OR LDCLF INSTRUCTION IS EXECUTED.
20429  ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
20430  ;COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCFSUB RETURNS CONTROL
20431  ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCFSUB WILL
20432  ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCFSUB WILL RETURN
20433  ;TO THE ERROR CALL AT ERR2, OTHERWISE LDCFSUB ITSELF
20434  ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
20435  ;LDCIF OR LDCLF IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
20436  ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
20437  ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDCFSUB
20438  ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
20439  ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCFSUB
     ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
```

```
20440
20441  112404  012601                  LDCFSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
20442  112406  016100  000014                  MOV     14(R1),R0       ;SET THE FPS.
20443  112412  170100                          LDFPS   R0
20444  112414  010100                          MOV     R1,R0
20445
20446  112416  177010                  1$:     LDCIF   (R0),AC0                ;TEST INSTRUCTION LDCIF OR LDCLF.
20447
20448  112420  170204                          STFPS   R4              ;GET FPS.
20449  112422  012700  112476                  MOV     #LDCT,R0        ;GET THE RESULT.
20450  112426  012702  000200                  MOV     #200,R2
20451  112432  170102                          LDFPS   R2
20452  112434  174010                          STD     AC0,(R0)
20453
20454  112436  012702  112476                  MOV     #LDCT,R2        ;SEE IF THE RESULT WAS CORRECT.
20455  112442  010100                          MOV     R1,R0
20456  112444  062700  000004                  ADD     #4,R0
20457  112450  012703  000002                  MOV     #2,R3
20458  112454  022022                  2$:     CMP     (R0)+,(R2)+
20459  112456  001006                          BNE     10$             ;BR IF INCORRECT.
20460  112460  077303                          SOB     R3,2$
20461
20462  112462  026104  000016                  CMP     16(R1),R4       ;SEE IF THE FPS WAS CORRECT.
20463  112466  001002                          BNE     10$             ;BR IF INCORRECT.
20464  112470  000161  000022          3$:     JMP     22(R1)          ;RETURN.
20465  112474                          10$:
20466  112474  104000                          EMT                     ;
20467
20468                                  ;DATA BUFFER:
20469  112476  000000  000000  000000  LDCT:   .WORD   0,0,0,0
20470  112504  000000
20471
20472  112506                          KKCDONE:
20473  112506  004767  005466                  JSR     PC,,RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20474                                                                  ;SEE IF THE USER HAS EXPRESSED
20475                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
20476                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20477                                                                  ;THE USER TYPED CONTROL G?).
20478
20479
20480                                  ;*****************************************************************************************
20481                                  ;TEST 556       LDCID AND LDCLD TEST
20482                                  ;*****************************************************************************************
20483  112512                          TS556:
20484                                  ;OPERAND=0      FL=0,   FD=1
20485  112512  004737  113170          LLC1:   JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
20486  112516  000000  000000          1$:     .WORD   0,0             ;FSRC OPERAND.
20487  112522  000000  000000  000000  2$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
20488  112530  000000
20489  112532  177777  177777  177777  3$:     .WORD   -1,-1,-1,-1             ;ANTICIPATED ERRONEOUS RESULT.
20490  112540  177777
20491  112542  000213                  4$:     213                     ;FPS BEFORE EXECUTION.
20492  112544  000204                          204                     ;FPS AFTER EXECUTION.
20493  112546  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
20494                                  ;OPERAND=0      FL=0,   FD=1
20495  112550  004737  113170          LLC2:   JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
```

```
20496  112554  000000  177777                1$:     .WORD   0,-1                    ;FSRC OPERAND.
20497  112560  000000  000000  000000        2$:     .WORD   0,0,0,0                 ;EXPECTED RESULT.
20498  112566  000000
20499  112570  004177  177400  000000        3$:     .WORD   4177,177400,0,0                 ;ANTICIPATED ERRONEOUS RESULT.
20500  112576  000000
20501  112600  000200                        4$:     200                             ;FPS BEFORE EXECUTION.
20502  112602  000204                                204                             ;FPS AFTER EXECUTION.
20503  112604  177777                                -1                              ;ANTICIPATED ERRONEOUS FPS.
20504                                  ;OPERAND=0      FL=1    FD=1
20505  112606  004737  113170          LLC3:   JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
20506  112612  000000  000000          1$:     .WORD   0,0             ;FSRC OPERAND.
20507  112616  000000  000000  000000  2$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
20508  112624  000000
20509  112626  177777  177777  177777  3$:     .WORD   -1,-1,-1,-1             ;ANTICIPATED ERRONEOUS RESULT.
20510  112634  177777
20511  112636  000211                        4$:     211                             ;FPS BEFORE EXECUTION.
20512  112640  000204                                204                             ;FPS AFTER EXECUTION.
20513  112642  177777                                -1                              ;ANTICIPATED ERRONEOUS FPS.
20514                                  ;OPERAND=40000  FL=0    FD=1
20515  112644  004737  113170          LLC4:   JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
20516  112650  040000  000000          1$:     .WORD   40000,0         ;FSRC OPERAND.
20517  112654  043600  000000  000000  2$:     .WORD   43600,0,0,0             ;EXPECTED RESULT.
20518  112662  000000
20519  112664  047600  000000  000000  3$:     .WORD   47600,0,0,0             ;ANTICIPATED ERRONEOUS RESULT.
20520  112672  000000
20521  112674  000217                        4$:     217                             ;FPS BEFORE EXECUTION.
20522  112676  000200                                200                             ;FPS AFTER EXECUTION.
20523  112700  177777                                -1                              ;ANTICIPATED ERRONEOUS FPS.
20524                                  ;OPERAND=-40000 FL=0    FD=1
20525  112702  004737  113170          LLC5:   JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
20526  112706  140000  000000          1$:     .WORD   -40000,0                ;FSRC OPERAND.
20527  112712  143600  000000  000000  2$:     .WORD   143600,0,0,0            ;EXPECTED RESULT.
20528  112720  000000
20529  112722  043600  000000  000000  3$:     .WORD   43600,0,0,0                     ;ANTICIPATED ERRONEOUS RESULT.
20530  112730  000000
20531  112732  000200                        4$:     200                             ;FPS BEFORE EXECUTION.
20532  112734  000210                                210                             ;FPS AFTER EXECUTION.
20533  112736  177777                                -1                              ;ANTICIPATED ERRONEOUS FPS.
20534                                  ;OPERAND=40000,0        FL=1    FD=1
20535  112740  004737  113170          LLC6:   JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
20536  112744  040000  000000          1$:     .WORD   40000,0         ;FSRC OPERAND.
20537  112750  047600  000000  000000  2$:     .WORD   47600,0,0,0             ;EXPECTED RESULT.
20538  112756  000000
20539  112760  043600  000000  000000  3$:     .WORD   43600,0,0,0             ;ANTICIPATED ERRONEOUS RESULT.
20540  112766  000000
20541  112770  000317                        4$:     317                             ;FPS BEFORE EXECUTION.
20542  112772  000300                                300                             ;FPS AFTER EXECUTION.
20543  112774  177777                                -1                              ;ANTICIPATED ERRONEOUS FPS.
20544                                  ;OPERAND=0,1    FL=1    FD=1
20545  112776  004737  113170          LLC7:   JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
20546  113002  000000  000001          1$:     .WORD   0,1             ;FSRC OPERAND.
20547  113006  040200  000000  000000  2$:     .WORD   40200,0,0,0             ;EXPECTED RESULT.
20548  113014  000000
20549  113016  034200  000000  000000  3$:     .WORD   34200,0,0,0             ;ANTICIPATED ERRONEOUS RESULT.
20550  113024  000000
20551  113026  000300                        4$:     300                             ;FPS BEFORE EXECUTION.
```

```
20552  113030  000300                          300                     ;FPS AFTER EXECUTION.
20553  113032  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
20554                                  ;OPERAND=77777,177777   FL=1     FD=1
20555  113034  004737  113170          LLC10:  JSR     PC,@#LDCDSUB    ;GO EXECUTE THE INSTRUCTION.
20556  113040  077777  177777          1$:     .WORD   77777,177777            ;FSRC OPERAND.
20557  113044  047777  177777  177000  2$:     .WORD   47777,177777,177000,0           ;EXPECTED RESULT.
20558  113052  000000
20559  113054  177777  177777  177777  3$:     .WORD   -1,-1,-1,-1             ;ANTICIPATED ERRONEOUS RESULT.
20560  113062  177777
20561  113064  000317                  4$:     317                     ;FPS BEFORE EXECUTION.
20562  113066  000300                          300                     ;FPS AFTER EXECUTION.
20563  113070  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
20564                                  ;OPERAND=-PATTERN       FL=1    FD=1
20565
20566  113072  004767  000072          LLC11:  JSR     PC,LDCDSUB      ;GO EXECUTE THE INSTRUCTION.
20567  113076  177777  177526          1$:     .WORD   -1,-252         ;FSRC OPERAND.
20568  113102  142052  000000  000000  2$:     .WORD   142052,0,0,0            ;EXPECTED RESULT.
20569  113110  000000
20570  113112  136052  000000  000000  3$:     .WORD   136052,0,0,0            ;ANTICIPATED ERRONEOUS RESULT.
20571  113120  000000
20572  113122  000307                  4$:     307                     ;FPS BEFORE EXECUTION.
20573  113124  000310                          310                     ;FPS AFTER EXECUTION.
20574  113126  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
20575                                  ;OPERAND=PATTERN        FL=1    FD=1    FT=1
20576  113130  004767  000034          LLC12:  JSR     PC,LDCDSUB      ;GO EXECUTE THE INSTRUCTION.
20577  113134  012345  067012          1$:     .WORD   12345,67012             ;FSRC OPERAND.
20578  113140  047247  025560  050000  2$:     .WORD   47247,025560,050000,0           ;EXPECTED RESULT.
20579  113146  000000
20580  113150  177777  177777  177777  3$:     .WORD   -1,-1,-1,-1             ;ANTICIPATED ERRONEOUS RESULT.
20581  113156  177777
20582  113160  000352                  4$:     352                     ;FPS BEFORE EXECUTION.
20583  113162  000340                          340                     ;FPS AFTER EXECUTION.
20584  113164  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
20585  113166  000435                  6$:     BR      LLCDONE
20586
20587                                  ;THIS SUBROUTINE, LDCDSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
20588                                  ;THE LDCID OR LDCLD INSTRUCTION AND CHECK THE RESULTS. A CALL
20589                                  ;TO IT IS MADE THUS:
20590                                  ;
20591                                  ;               JSR     PC,@#LDCDSUB
20592                                  ;               ACARG:  .WORD   X,X             ;AC OPERAND
20593                                  ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
20594                                  ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
20595                                  ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
20596                                  ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
20597                                  ;               ERFPS:  .WORD   X               ;ERROR FPS.
20598                                  ;               ERR1:   ERROR   X               ;DATA ERROR.
20599                                  ;                       BR      CONT
20600                                  ;               ERR2:   ERROR   X               ;FPS ERROR.
20601                                  ;               CONT:                           ;RETURN ADDRESS
20602                                  ;
20603                                  ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
20604                                  ;THE LDCID OR LDCLD INSTRUCTION IS EXECUTED.
20605                                  ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
20606                                  ;COMPARED WITH FPSA IF THIS TOO IS CORRECT LDCDSUB RETURNS CONTROL
20607                                  ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDCDSUB
```

```
20608                                   ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDCDSUB WILL RETURN
20609                                   ;TO THE ERROR CALL AT ERR2, OTHERWISE LDCDSUB ITSELF
20610                                   ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
20611                                   ;LDCID OR LDCLD IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
20612                                   ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
20613                                   ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDCDSUB
20614                                   ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
20615                                   ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDCDSUB WILL
20616                                   ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
20617
20618  113170  012601          LDCDSUB:        MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
20619  113172  016100  000024                  MOV     24(R1),R0       ;SET THE FPS.
20620  113176  170100                          LDFPS   R0
20621  113200  010100                          MOV     R1,R0
20622  113202  177010          1$:             LDCID   (R0),AC0        ;TEST INSTRUCTION, LDCID OR LDCLD.
20623
20624  113204  170204                          STFPS   R4              ;GET FPS.
20625  113206  012700  112476                  MOV     #LDCT,R0        ;GET THE RESULT.
20626  113212  012702  000200                  MOV     #200,R2
20627  113216  170102                          LDFPS   R2
20628  113220  174010                          STD     AC0,(R0)
20629
20630                                   ;SEE IF THE RESULT IS CORRECT.
20631  113222  012702  112476                  MOV     #LDCT,R2
20632  113226  010100                          MOV     R1,R0
20633  113230  062700  000004                  ADD     #4,R0
20634  113234  012703  000002                  MOV     #2,R3
20635  113240  022022          2$:             CMP     (R0)+,(R2)+
20636  113242  001006                          BNE     10$             ;BR IF INCORRECT.
20637  113244  077303                          SOB     R3,2$
20638
20639  113246  026104  000026                  CMP     26(R1),R4               ;IS THE FPS CORRECT?
20640  113252  001002                          BNE     10$             ;BR IF INCORRECT.
20641  113254  000161  000032          3$:     JMP     32(R1)          ;RETURN.
20642  113260                          10$:
20643  113260  104000                          EMT                     ;
20644
20645  113262                          LLCDONE:
20646  113262  004767  004712                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
20647                                                                  ;SEE IF THE USER HAS EXPRESSED
20648                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
20649                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20650                                                                  ;THE USER TYPED CONTROL G?).
20651
20652
20653                                   ;*****************************************************************************
20654                                   ;TEST 557       LDEXP TEST
20655                                   ;*****************************************************************************
20656  113266                          TS557:
20657
20658                                   ; NON-ZERO RES. VALID EXPON=210 (EXCESS 200)=10
20659  113266  004767  001136          MMC1:   JSR     PC,LDXSUB       ;GO EXECUTE THE INSTRUCTION.
20660  113272  012345  067012  034567  1$:     .WORD   12345,67012,34567,012345        ;AC0 OPERAND.
20661  113300  012345
20662  113302  000010                  2$:     .WORD   10              ;EXPONENT OPERAND.
20663  113304  042145  067012  034567  3$:     .WORD   42145,67012,34567,012345                ;EXPECTED RESULT.
```

```
20664  113312  012345
20665  113314  002145  067012  034567  4$:     .WORD   2145,67012,34567,012345          ;ANTICIPATED ERRONEOUS RESULT.
20666  113322  012345
20667  113324  047217                  5$:     47217                   ;FPS BEFORE EXECUTION.
20668  113326  047200                          47200                   ;FPS AFTER EXECUTION.
20669  113330  147200                          147200                  ;ANTICIPATED ERRONEOUS FPS.
20670  113332  177777                          -1              ;EXPECTED FEC.
20671                                  ;NON-ZERO       RES     NEG.
20672  113334  004737  114430          MMC2:   JSR     PC,@#LDXSUB     ;EXPON=377
20673  113340  123456  070123  045670  1$:     .WORD   123456,70123,45670,123456       ;AC0 OPERAND.
20674  113346  123456
20675  113350  000177                  2$:     .WORD   177             ;EXPONENT OPERAND.
20676  113352  177656  070123  045670  3$:     .WORD   177656,70123,45670,123456               ;EXPECTED RESULT.
20677  113360  123456
20678  113362  137656  070123  045670  4$:     .WORD   137656,70123,45670,123456               ;ANTICIPATED ERRONEOUS RESULT.
20679  113370  123456
20680  113372  047207                  5$:     47207                   ;FPS BEFORE EXECUTION.
20681  113374  047210                          47210                   ;FPS AFTER EXECUTION.
20682  113376  147210                          147210                  ;ANTICIPATED ERRONEOUS FPS.
20683  113400  177777                          -1              ;EXPECTED FEC.
20684                                  ;NON-ZERO       RES,    EXP=256=(56)REAL
20685  113402  004737  114430          MMC3:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20686  113406  073261  057645  043323  1$:     .WORD   73261,057645,43323,101760       ;AC0 OPERAND.
20687  113414  101760
20688  113416  000056                  2$:     .WORD   56              ;EXPONENT OPERAND.
20689  113420  053461  057645  043323  3$:     .WORD   53461,057645,43323,101760               ;EXPECTED RESULT.
20690  113426  101760
20691  113430  177777  177777  177777  4$:     .WORD   -1,-1,-1,-1                     ;ANTICIPATED ERRONEOUS RESULT.
20692  113436  177777
20693  113440  047200                  5$:     47200                   ;FPS BEFORE EXECUTION.
20694  113442  047200                          47200                   ;FPS AFTER EXECUTION.
20695  113444  147200                          147200                  ;ANTICIPATED ERRONEOUS FPS.
20696  113446  177777                          -1              ;EXPECTED FEC.
20697                                  ;EXP=27 (EXCESS 200)=-151 (OCT)
20698  113450  004737  114430          MMC4:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20699  113454  012223  024252  062720  1$:     .WORD   12223,24252,62720,21222 ;AC0 OPERAND.
20700  113462  021222
20701  113464  177627                  2$:     .WORD   -151            ;EXPONENT OPERAND.
20702  113466  005623  024252  062720  3$:     .WORD   5623,24252,62720,21222          ;EXPECTED RESULT.
20703  113474  021222
20704  113476  177777  177777  177777  4$:     .WORD   -1,-1,-1,-1                     ;ANTICIPATED ERRONEOUS RESULT.
20705  113504  177777
20706  113506  047200                  5$:     47200                   ;FPS BEFORE EXECUTION.
20707  113510  047200                          47200                   ;FPS AFTER EXECUTION.
20708  113512  147200                          147200                  ;ANTICIPATED ERRONEOUS FPS.
20709  113514  177777                          -1              ;EXPECTED FEC.
20710                                  ;EXP=0 (EXCESS 200)=-200 (OCT), POSITIVE FRAC
20711                                  ; FIV=1
20712  113516  004737  114430          MMC5:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20713  113522  030131  032334  035363  1$:     .WORD   30131,32334,35363,73031 ;AC0 OPERAND.
20714  113530  073031
20715  113532  177600                  2$:     .WORD   -200            ;EXPONENT OPERAND.
20716  113534  000131  032334  035363  3$:     .WORD   00131,32334,35363,73031         ;EXPECTED RESULT.
20717  113542  073031
20718  113544  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
20719  113552  000000
```

```
20720  113554  042200                  5$:     42200                   ;FPS BEFORE EXECUTION.
20721  113556  142204                          142204                  ;FPS AFTER EXECUTION.
20722  113560  042202                          42202                   ;ANTICIPATED ERRONEOUS FPS.
20723  113562  000012                          12              ;EXPECTED FEC.
20724                                  ;EXP=0  (EXCESS 200)=-200 (OCT), NEG FRACT,FIU=1
20725  113564  004737  114430          MMC6:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20726  113570  140414  024344  045464  1$:     .WORD   140414,24344,45464,74045        ;AC0 OPERAND.
20727  113576  074045
20728  113600  177600                  2$:     .WORD   -200            ;EXPONENT OPERAND.
20729  113602  100014  024344  045464  3$:     .WORD   100014,24344,45464,74045        ;-0             ;EXPECTED RESULT.
20730  113610  074045
20731  113612  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
20732  113620  000000
20733  113622  042200                  5$:     42200                   ;FPS BEFORE EXECUTION.
20734  113624  142214                          142214                  ;FPS AFTER EXECUTION.
20735  113626  042214                          42214                   ;ANTICIPATED ERRONEOUS FPS.
20736  113630  000012                          12              ;EXPECTED FEC.
20737                                  ;EXP=0 (EXCESS 200)=-200 (OCT),POS FRAC, FIU=0
20738
20739  113632  004737  114430          MMC7:   JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20740  113636  051525  035455  005675  1$:     .WORD   51525,35455,5675,05152  ;AC0 OPERAND.
20741  113644  005152
20742  113646  177600                  2$:     .WORD   -200            ;EXPONENT OPERAND.
20743  113650  000000  000000  000000  3$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
20744  113656  000000
20745  113660  000125  035455  005675  4$:     .WORD   00125,35455,5675,05152          ;ANTICIPATED ERRONEOUS RESULT.
20746  113666  005152
20747  113670  045200                          45200                   ;FPS BEFORE EXECUTION.
20748  113672  045204                          45204                   ;FPS AFTER EXECUTION.
20749  113674  145204                          145204                  ;ANTICIPATED ERRONEOUS FPS.
20750  113676  177777                          -1              ;EXPECTED FEC.
20751                                  ;EXP=-1405 (EXCESS 200)=-1605 (OCT), FIU=1
20752  113700  004737  114430          MMC10:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20753  113704  061626  062636  046566  1$:     .WORD   61626,62636,46566,67606 ;AC0 OPERAND.
20754  113712  067606
20755  113714  176173                  2$:     .WORD   -1605           ;EXPONENT OPERAND.
20756  113716  076626  062636  046566  3$:     .WORD   76626,62636,46566,67606         ;EXPECTED RESULT.
20757  113724  067606
20758  113726  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
20759  113734  000000
20760  113736  042200                  5$:     42200                   ;FPS BEFORE EXECUTION.
20761  113740  142200                          142200                  ;FPS AFTER EXECUTION.
20762  113742  042204                          42204                   ;ANTICIPATED ERRONEOUS FPS.
20763  113744  000012                          12              ;EXPECTED FEC.
20764                                  ;EXP=-17416 (EXCESS 200)=-17616 (OCT), FIU=0
20765  113746  004737  114430          MMC11:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20766  113752  071727  037475  076777  1$:     .WORD   71727,37475,76777,17273 ;AC0 OPERAND.
20767  113760  017273
20768  113762  160162                  2$:     .WORD   -17616          ;EXPONENT OPERAND.
20769  113764  000000  000000  000000  3$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
20770  113772  000000
20771  113774  074527  037475  076777  4$:     .WORD   74527,37475,76777,17273         ;ANTICIPATED ERRONEOUS RESULT.
20772  114002  017273
20773  114004  045200                  5$:     45200                   ;FPS BEFORE EXECUTION.
20774  114006  045204                          45204                   ;FPS AFTER EXECUTION.
20775  114010  145200                          145200                  ;ANTICIPATED ERRONEOUS FPS.
```

```
20776  114012  177777                          -1                      ;EXPECTED FEC.
20777                                  ;EXP=-1601 (EXCESS 200)=-2001 (OCT), FIU=1
20778  114014  004737  114430          MMC12:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20779  114020  001020  030405  006070  1$:     .WORD   01020,30405,06070,00102 ;AC0 OPERAND.
20780  114026  000102
20781  114030  175777                  2$:     .WORD   -2001           ;EXPONENT OPERAND.
20782  114032  037620  030405  006070  3$:     .WORD   37620,30405,06070,00102         ;EXPECTED RESULT.
20783  114040  000102
20784  114042  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
20785  114050  000000
20786  114052  042200                  5$:     42200                   ;FPS BEFORE EXECUTION.
20787  114054  142200                          142200                  ;FPS AFTER EXECUTION.
20788  114056  042204                          42204                   ;ANTICIPATED ERRONEOUS FPS.
20789  114060  000012                          12                      ;EXPECTED FEC.
20790                                  ;EXP=1206 (EXCESS 200)=1006 (OCT) FIV =1
20791  114062  004737  114430          MMC13:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20792  114066  012131  014151  016171  1$:     .WORD   12131,14151,16171,10111 ;AC0 OPERAND.
20793  114074  010111
20794  114076  001006                  2$:     .WORD   1006            ;EXPONENT OPERAND.
20795  114100  041531  014151  016171  3$:     .WORD   41531,14151,16171,10111         ;EXPECTED RESULT.
20796  114106  010111
20797  114110  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
20798  114116  000000
20799  114120  041200                  5$:     41200                   ;FPS BEFORE EXECUTION.
20800  114122  141202                          141202                  ;FPS AFTER EXECUTION.
20801  114124  041204                          41204                   ;ANTICIPATED ERRONEOUS FPS.
20802  114126  000010                          10                      ;EXPECTED FEC.
20803                                  ;EXP=16315 (EXCESS 200)=16115 (OCT) FIV=0
20804  114130  004737  114430          MMC14:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20805  114134  027262  025242  023222  1$:     .WORD   27262,25242,23222,21202 ;AC0 OPERAND.
20806  114142  021202
20807  114144  016115                  2$:     .WORD   16115           ;EXPONENT OPERAND.
20808  114146  000000  000000  000000  3$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
20809  114154  000000
20810  114156  063262  025242  023222  4$:     .WORD   63262,25242,23222,21202         ;ANTICIPATED ERRONEOUS RESULT.
20811  114164  021202
20812  114166  046200                  5$:     46200                   ;FPS BEFORE EXECUTION.
20813  114170  046206                          46206                   ;FPS AFTER EXECUTION.
20814  114172  146202                          146202                  ;ANTICIPATED ERRONEOUS FPS.
20815  114174  177777                          -1                      ;EXPECTED FEC.
20816                                  ;EXP=11011 (EXCESS 200)=10611 (OCT) FIV=1
20817
20818  114176  004737  114430          MMC15:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20819  114202  030313  032333  034353  1$:     .WORD   30313,32333,34353,36373 ;AC0 OPERAND.
20820  114210  036373
20821  114212  010611                  2$:     .WORD   10611           ;EXPONENT OPERAND.
20822  114214  002313  032333  034353  3$:     .WORD   2313,32333,34353,36373          ;EXPECTED RESULT.
20823  114222  036373
20824  114224  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
20825  114232  000000
20826  114234  041200                  5$:     41200                   ;FPS BEFORE EXECUTION.
20827  114236  141202                          141202                  ;FPS AFTER EXECUTION.
20828  114240  041204                          41204                   ;ANTICIPATED ERRONEOUS FPS.
20829  114242  000010                          10                      ;EXPECTED FEC.
20830                                  ;EXP=17123 (EXCESS 200)=16723 (OCT) FIV=0
20831
```

```
20832  114244  004737  114430          MMC16:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20833  114250  040414  042434  044454  1$:     .WORD   40414,42434,44454,46474 ;AC0 OPERAND.
20834  114256  046474
20835  114260  016723                  2$:     .WORD   16723           ;EXPONENT OPERAND.
20836  114262  000000  000000  000000  3$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
20837  114270  000000
20838  114272  024614  042434  044454  4$:     .WORD   24614,42434,44454,46474         ;ANTICIPATED ERRONEOUS RESULT.
20839  114300  046474
20840  114302  046200                  5$:     46200                   ;FPS BEFORE EXECUTION.
20841  114304  046206                          46206                   ;FPS AFTER EXECUTION.
20842  114306  146202                          146202                  ;ANTICIPATED ERRONEOUS FPS.
20843  114310  177777                          -1              ;EXPECTED FEC.
20844                                  ;EXP=   254 (OCT)=      454 (EXCESS 200)        FIV=1
20845
20846  114312  004737  114430          MMC17:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20847  114316  050515  052535  054555  1$:     .WORD   50515,52535,54555,56575 ;AC0 OPERAND.
20848  114324  056575
20849  114326  000254                  2$:     .WORD   254             ;EXPONENT OPERAND.
20850  114330  013115  052535  054555  3$:     .WORD   13115,52535,54555,56575         ;EXPECTED RESULT.
20851  114336  056575
20852  114340  000000  000000  000000  4$:     .WORD   0,0,0,0         ;ANTICIPATED ERRONEOUS RESULT.
20853  114346  000000
20854  114350  041200                  5$:     41200                   ;FPS BEFORE EXECUTION.
20855  114352  141202                          141202                  ;FPS AFTER EXECUTION.
20856  114354  041204                          41204                   ;ANTICIPATED ERRONEOUS FPS.
20857  114356  000010                          10              ;EXPECTED FEC.
20858                                  ;EXP=   313 (OCT)=      513(EXCESS 200) FIV=0
20859
20860  114360  004737  114430          MMC20:  JSR     PC,@#LDXSUB     ;GO EXECUTE THE INSTRUCTION.
20861  114364  060616  062636  064656  1$:     .WORD   60616,62636,64656,66676 ;AC0 OPERAND.
20862  114372  066676
20863  114374  000313                  2$:     .WORD   313             ;EXPONENT OPERAND.
20864  114376  000000  000000  000000  3$:     .WORD   0,0,0,0         ;EXPECTED RESULT.
20865  114404  000000
20866  114406  022616  062636  064656  4$:     .WORD   22616,62636,64656,66676         ;ANTICIPATED ERRONEOUS RESULT.
20867  114414  066676
20868  114416  046200                  5$:     46200                   ;FPS BEFORE EXECUTION.
20869  114420  046206                          46206                   ;FPS AFTER EXECUTION.
20870  114422  146202                          146202                  ;ANTICIPATED ERRONEOUS FPS.
20871  114424  177777                          -1              ;EXPECTED FEC.
20872  114426  000457                          BR      MMCDONE
20873
20874                                  ;THIS SUBROUTINE, LDXSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
20875                                  ;THE LDEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
20876                                  ;TO IT IS MADE THUS:
20877                                  ;
20878                                  ;               JSR     PC,@#LDXSUB
20879                                  ;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
20880                                  ;               EXP:    .WORD   X               ;EXPONENT
20881                                  ;               RES:    .WORD   X,X,X,X         ;EXPECTED RESULT
20882                                  ;               ERRES:  .WORD   X,X,X,X         ;ERROR RESULT
20883                                  ;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
20884                                  ;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
20885                                  ;               ERFPS:  .WORD   X               ;ERROR FPS.
20886                                  ;               FEC:    .WORD   X               ;EXPECTED FEC
20887                                  ;               ERR1:   ERROR   X               ;DATA ERROR.
```

```
                                        BR      CONT
                                ERR2:   ERROR   X               ;FPS ERROR.
                                CONT:                           ;RETURN ADDRESS
                                ;
                                ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
                                ;THE LDEXP INSTRUCTION IS EXECUTED.
                                ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
                                ;COMPARED WITH FPSA IF THIS TOO IS CORRECT LDXSUB RETURNS CONTROL
                                ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD LDXSUB
                                ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN LDXSUB WILL RETURN
                                ;TO THE ERROR CALL AT ERR2, OTHERWISE LDXSUB ITSELF
                                ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
                                ;LDEXP IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
                                ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
                                ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN LDXSUB
                                ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
                                ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND LDXSUB WILL
                                ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.

20907  114430  012601          LDXSUB: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
20908  114432  012700  000200          MOV     #200,R0         ;LOAD THE AC0 OPERAND.
20909  114436  170100                  LDFPS   R0
20910  114440  010100                  MOV     R1,R0
20911  114442  172410                  LDD     (R0),AC0
20912  114444  016100  000032          MOV     32(R1),R0       ;SET UP THE FPS.
20913  114450  170100                  LDFPS   R0
20914  114452  010100                  MOV     R1,R0
20915  114454  062700  000010          ADD     #10,R0
20916
20917  114460  176410          1$:     LDEXP   (R0),AC0        ;TEST INSTRUCTION.
20918
20919  114462  170204                  STFPS   R4              ;GET THE FPS.
20920  114464  170305                  STST    R5              ;GET THE FEC.
20921  114466  012700  000200          MOV     #200,R0         ;GET THE RESULT.
20922  114472  170100                  LDFPS   R0
20923  114474  012700  114556          MOV     #LDXT,R0
20924  114500  174010                  STD     AC0,(R0)
20925  114502  012702  114556          MOV     #LDXT,R2        ;SEE IF THE RESULT WAS CORRECT.
20926  114506  010103                  MOV     R1,R3
20927  114510  062703  000012          ADD     #12,R3
20928  114514  012700  000004          MOV     #4,R0
20929  114520  022223          2$:     CMP     (R2)+,(R3)+
20930  114522  001014                  BNE     10$             ;BRANCH IF NOT CORRECT.
20931  114524  077003                  SOB     R0,2$
20932  114526  020461  000034          CMP     R4,34(R1)       ;SEE IF THE FPS WAS CORRECT.
20933  114532  001010                  BNE     10$             ;BRANCH IF NOT CORRECT.
20934  114534  005761  000034          TST     34(R1)
20935  114540  100003                  BPL     3$
20936  114542  020561  000040          CMP     R5,40(R1)       ;SEE IF THE FEC WAS CORRECT.
20937  114546  001002                  BNE     10$             ;BRANCH IF NOT CORRECT.
20938
20939  114550  000161  000042  3$:     JMP     42(R1)          ;RETURN.
20940  114554                  10$:
20941  114554  104000                  EMT                     ;
20942
20943                          ;DATA BUFFER:
```

```
20944  114556  000000  000C00  000000  LDXT:   .WORD   0,0,0,0
20945  114564  000000
20946
20947  114566                          MMCDONE:
20948  114566  004767  003406                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20949                                                                  ;SEE IF THE USER HAS EXPRESSED
20950                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
20951                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20952                                                                  ;THE USER TYPED CONTROL G?).
20953
20954
20955
20956                                  ;*****************************************************************************
20957                                  ;TEST 560       DESTINATION MODES, MODE 1 (FL=0), TEST
20958                                  ;*****************************************************************************
20959  114572                          TS560:
20960
20961
20962  114572  012700  114662                  MOV     #NNCTB0,R0      ;SET UP THE DATA BUFFER.
20963  114576  012701  000006                  MOV     #6,R1
20964  114602  012720  177777          1$:     MOV     #-1,(R0)+
20965  114606  077103                          SOB     R1,1$
20966  114610  012700  102345                  MOV     #102345,R0
20967  114614  012737  114676  000004        MOV     #NNC10,@#ERRVECT ;SET UP FOR TRAPS TO 4.
20968  114622  170100                          LDFPS   R0              ;SET UP FPS.
20969  114624  012700  114666                  MOV     #NNCTB1,R0
20970
20971  114630  170210                  NNC2:   STFPS   (R0)            ;TEST INSTRUCTION.
20972  114632  020027  114666                  CMP     R0,#NNCTB1      ;IS R0 CORRECT?
20973  114636  001017                          BNE     NNC10           ;BRANCH IF NOT CORRECT.
20974  114640  023727  114666  102345        CMP     @#NNCTB1,#102345   ;IS RESULT CORRECT?
20975  114646  001013                          BNE     NNC10           ;BRANCH IF NOT CORRECT.
20976  114650  023727  114670  177777        CMP     @#NNCTB1+2,#-1  ;IS THE RESULT CORRECT?
20977  114656  001007                          BNE     NNC10           ;BRANCH IF NOT CORRECT.
20978  114660  000407                          BR      NNCDONE
20979
20980                                  ;TEST DATA BUFFER:
20981  114662  177777  177777          NNCTB0: .WORD   -1,-1
20982  114666  177777  177777  177777  NNCTB1: .WORD   -1,-1,-1,-1
20983  114674  177777
20984  114676                          NNC10:
20985  114676  104000                          EMT                     ;
20986
20987  114700                          NNCDONE:
20988  114700  004767  003274                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
20989                                                                  ;SEE IF THE USER HAS EXPRESSED
20990                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
20991                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
20992                                                                  ;THE USER TYPED CONTROL G?).
20993
20994
20995                                  ;*****************************************************************************
20996                                  ;TEST 561       DESTINATION MODES, MODE 2 (FL=0), TEST
20997                                  ;*****************************************************************************
20998  114704                          TS561:
20999
```

```
21000
21001  114704  012700  114774                  MOV     #OOCTB0,R0      ;SET UP THE DATA BUFFER.
21002  114710  012701  000006                  MOV     #6,R1
21003  114714  012720  177777          1$:     MOV     #-1,(R0)+
21004  114720  077103                          SOB     R1,1$
21005  114722  012700  105412                  MOV     #105412,R0
21006  114726  012737  115010  000004          MOV     #OOC10,@#ERRVECT  ;SET UP FOR TRAPS TO VECTOR 4.
21007  114734  170100                          LDFPS   R0              ;SET UP FPS.
21008  114736  012700  115000                  MOV     #OOCTB1,R0
21009
21010  114742  170220                  OOC2:   STFPS   (R0)+           ;TEST INSTRUCTION.
21011  114744  020027  115002                  CMP     R0,#OOCTB1+2    ;IS R0 CORRECT?
21012  114750  001017                          BNE     OOC10           ;BRANCH IF NOT CORRECT.
21013  114752  023727  115000  105412          CMP     @#OOCTB1,#105412  ;IS THE RESULT CORRECT?
21014  114760  001013                          BNE     OOC10           ;BRANCH IF NOT CORRECT.
21015  114762  023727  115002  177777          CMP     @#OOCTB1+2,#-1  ;IS THE RESULT CORRECT?
21016  114770  001007                          BNE     OOC10           ;BRANCH IF NOT CORRECT.
21017  114772  000407                          BR      OOCDONE
21018
21019                                  ;TEST DATA BUFFER:
21020  114774  177777  177777          OOCTB0: .WORD   -1,-1
21021  115000  177777  177777  177777  OOCTB1: .WORD   -1,-1,-1,-1
21022  115006  177777
21023  115010                          OOC10:
21024  115010  104000                          EMT                     ;
21025
21026  115012                          OOCDONE:
21027  115012  004767  003162                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
21028                                                                  ;SEE IF THE USER HAS EXPRESSED
21029                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
21030                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21031                                                                  ;THE USER TYPED CONTROL G?).
21032
21033
21034
21035                                  ;*****************************************************************************
21036                                  ;TEST 562       DESTINATION MODES, MODE 4 (FL=0), TEST
21037                                  ;*****************************************************************************
21038  115016                          TS562:
21039
21040  115016  012700  115106                  MOV     #PPCTB0,R0      ;SET UP THE DATA BUFFER.
21041  115022  012701  000006                  MOV     #6,R1
21042  115026  012720  177777          1$:     MOV     #-1,(R0)+
21043  115032  077103                          SOB     R1,1$
21044  115034  012700  105555                  MOV     #105555,R0
21045  115040  012737  115122  000004          MOV     #PPC10,@#ERRVECT  ;SET UP FOR TRAPS TO VECTOR 4.
21046  115046  170100                          LDFPS   R0              ;SET UP FPS.
21047  115050  012700  115114                  MOV     #PPCTB1+2,R0
21048
21049  115054  170240                  PPC2:   STFPS   -(R0)           ;TEST INSTRUCTION.
21050  115056  020027  115112                  CMP     R0,#PPCTB1      ;IS R0 CORRECT?
21051  115062  001017                          BNE     PPC10           ;BRANCH IF NOT CORRECT.
21052  115064  023727  115112  105555          CMP     @#PPCTB1,#105555  ;IS THE RESULT CORRECT?
21053  115072  001013                          BNE     PPC10           ;BRANCH IF NOT CORRECT.
21054  115074  023727  115114  177777          CMP     @#PPCTB1+2,#-1  ;IS THE RESULT CORRECT?
21055  115102  001007                          BNE     PPC10           ;BRANCH IF NOT CORRECT.
```

```
21056  115104  000407                          BR      PPCDONE
21057
21058                                          ;TEST DATA BUFFER:
21059  115106  177777  177777                 PPCTB0: .WORD   -1,-1
21060  115112  177777  177777  177777         PPCTB1: .WORD   -1,-1,-1,-1
21061  115120  177777
21062  115122                                 PPC10:
21063  115122  104000                          EMT                     ;
21064  115124                                 PPCDONE:
21065  115124  004767  003050                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
21066                                                                 ;SEE IF THE USER HAS EXPRESSED
21067                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
21068                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21069                                                                 ;THE USER TYPED CONTROL G?).
21070
21071
21072
21073                                          ;*****************************************************************************
21074                                          ;TEST 563       DESTINATION MODES, MODE 3 (FL=0), TEST
21075                                          ;*****************************************************************************
21076  115130                                 TS563:
21077
21078  115130  012700  115224                  MOV     #QQCTB0,R0      ;SET UP THE DATA BUFFER.
21079  115134  012701  000010                  MOV     #10,R1
21080  115140  012720  177777          1$:     MOV     #-1,(R0)+
21081  115144  077103                          SOB     R1,1$
21082  115146  012700  106653                  MOV     #106653,R0
21083  115152  012737  115244  000004          MOV     #QQC10,@#ERRVECT  ;SET UP FOR TRAPS TO VECTOR 4.
21084  115160  170100                          LDFPS   R0              ;SET UP FPS.
21085  115162  012700  115240                  MOV     #QQCTB2,R0
21086  115166  012710  115230                  MOV     #QQCTB1,(R0)
21087
21088  115172  170230                  QQC2:   STFPS   @(R0)+           ;TEST INSTRUCTION.
21089  115174  020027  115242                  CMP     R0,#QQCTB2+2    ;IS R0 CORRECT?
21090  115200  001021                          BNE     QQC10           ;BRANCH IF NOT CORRECT.
21091  115202  023727  115230  106653          CMP     @#QQCTB1,#106653  ;IS THE RESULT CORRECT?
21092  115210  001015                          BNE     QQC10           ;BRANCH IF NOT CORRECT.
21093  115212  023727  115240  115230          CMP     @#QQCTB2,@#QQCTB1  ;IS THE RESULT CORRECT?
21094  115220  001011                          BNE     QQC10           ;BRANCH IF NOT CORRECT.
21095  115222  000411                          BF      QQCDONE
21096
21097                                          ;TEST DATA BUFFER:
21098  115224  177777  177777                 QQCTB0: .WORD   -1,-1
21099  115230  177777  177777  177777         QQCTB1: .WORD   -1,-1,-1,-1
21100  115236  177777
21101  115240  177777  177777                 QQCTB2: .WORD   -1,-1
21102  115244                                 QQC10:
21103  115244  104000                          EMT                     ;
21104  115246                                 QQCDONE:
21105  115246  004767  002726                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
21106                                                                 ;SEE IF THE USER HAS EXPRESSED
21107                                                                 ;THE DESIRE TO CHANGE THE SOFTWARE
21108                                                                 ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21109                                                                 ;THE USER TYPED CONTROL G?).
21110
21111
```

```
21112
21113                                   ;*****************************************************************************
21114                                   ;TEST 564       DESTINATION MODES, MODE 5 (FL=0), TEST
21115                                   ;*****************************************************************************
21116  115252                          TS564:
21117
21118
21119  115252  012700  115350                  MOV     #RRCTB0,R0      ;SET UP THE DATA BUFFER.
21120  115256  012701  000006                  MOV     #6,R1
21121  115262  012720  177777          1$:     MOV     #-1,(R0)+
21122  115266  077103                          SOB     R1,1$
21123  115270  012700  004301                  MOV     #004301,R0
21124  115274  012737  115370  000004        MOV     #RRC10,@#ERRVECT  ;SET UP FOR TRAPS TO VECTOR 4.
21125  115302  170100                          LDFPS   R0              ;SET UP FPS.
21126  115304  012700  115366                  MOV     #RRCTB2+2,R0
21127  115310  012760  115354  177776        MOV     #RRCTB1,-2(R0)
21128
21129  115316  170250                  RRC2:   STFPS   @-(R0)          ;TEST INSTRUCTION.
21130  115320  020027  115364                  CMP     R0,#RRCTB2      ;IS R0 CORRECT?
21131  115324  001021                          BNE     RRC10           ;BRANCH IF NOT CORRECT.
21132  115326  023727  115354  004301        CMP     @#RRCTB1,#004301  ;IS THE RESULT CORRECT?
21133  115334  001015                          BNE     RRC10           ;BRANCH IF NOT CORRECT.
21134  115336  023727  115364  115354        CMP     @#RRCTB2,#RRCTB1  ;IS THE RESULT CORRECT?
21135  115344  001011                          BNE     RRC10           ;BRANCH IF NOT CORRECT.
21136  115346  000411                          BR      RRCDONE
21137
21138                                   ;TEST DATA BUFFER:
21139  115350  177777  177777          RRCTB0: .WORD   -1,-1
21140  115354  177777  177777  177777  RRCTB1: .WORD   -1,-1,-1,-1
21141  115362  177777
21142  115364  177777  177777          RRCTB2: .WORD   -1,-1
21143  115370                          RRC10:
21144  115370  104000                          EMT                     ;
21145  115372                          RRCDONE:
21146  115372  004767  002602                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
21147                                                                  ;SEE IF THE USER HAS EXPRESSED
21148                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
21149                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21150                                                                  ;THE USER TYPED CONTROL G?).
21151
21152
21153                                   ;*****************************************************************************
21154                                   ;TEST 565       DESTINATION MODES, MODE 6 (FL=0), TEST
21155                                   ;*****************************************************************************
21156  115376                          TS565:
21157
21158
21159  115376  012700  115500                  MOV     #SSCTB0,R0      ;SET UP THE DATA BUFFER.
21160  115402  012701  000006                  MOV     #6,R1
21161  115406  012720  177777          1$:     MOV     #-1,(R0)+
21162  115412  077103                          SOB     R1,1$
21163  115414  012700  102514                  MOV     #102514,R0
21164  115420  012737  115514  000004        MOV     #SSC10,@#ERRVECT  ;SET UP FOR TRAPS TO VECTOR 4.
21165  115426  170100                          LDFPS   R0              ;SET UP FPS.
21166  115430  005001                          CLR     R1
21167  115432  012700  110303                  MOV     #SSCTB1-5201,R0
```

```
21168
21169  115436  170260  005201          SSC2:   STFPS   5201(R0)                 ;TEST INSTRUCTION.
21170  115442  020127  000000                  CMP     R1,#0           ;WAS PC CORRECT AFTER EXECUTION?
21171  115446  001022                          BNE     SSC10           ;BRANCH IF NOT CORRECT.
21172  115450  020027  110303                  CMP     R0,#SSCTB1-5201 ;IS R0 CORRECT?
21173  115454  001017                          BNE     SSC10           ;BRANCH IF NOT CORRECT.
21174  115456  023727  115504  102514          CMP     @#SSCTB1,#102514  ;IS THE RESULT CORRECT?
21175  115464  001013                          BNE     SSC10           ;BRANCH IF NOT CORRECT.
21176  115466  023727  115506  177777          CMP     @#SSCTB1+2,#-1  ;IS THE RESULT CORRECT?
21177  115474  001007                          BNE     SSC10           ;BRANCH IF NOT CORRECT.
21178  115476  000407                          BR      SSCDONE
21179
21180                                  ;TEST DATA BUFFER:
21181  115500  177777  177777          SSCTB0: .WORD   -1,-1
21182  115504  177777  177777  177777  SSCTB1: .WORD   -1,-1,-1,-1
21183  115512  177777
21184  115514                          SSC10:
21185  115514  104000                          EMT                     ;
21186  115516                          SSCDONE:
21187  115516  004767  002456                  JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
21188                                                                  ;SEE IF THE USER HAS EXPRESSED
21189                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
21190                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21191                                                                  ;THE USER TYPED CONTROL G?).
```

```
21192
21193
21194                          ;*****************************************************************************
21195                          ;TEST 566       DESTINATION MODES, MODE 7 (FL=0), TEST
21196                          ;*****************************************************************************
21197  115522                  TS566:
21198
21199  115522  012700  115632          MOV     #TTCTB0,R0      ;SET UP THE DATA BUFFER.
21200  115526  012701  000010          MOV     #10,R1
21201  115532  012720  177777  1$:     MOV     #-1,(R0)+
21202  115536  077103                  SOB     R1,1$
21203  115540  012700  103747          MOV     #103747,R0
21204  115544  012737  115652  000004  MOV     #TTC10,@#ERRVECT  ;SET UP FOR TRAPS TO VECTOR 4.
21205  115552  170100                  LDFPS   R0              ;SET UP FPS.
21206  115554  005001                  CLR     R1
21207  115556  012700  110445          MOV     #TTCTB2-5201,R0
21208  115562  012760  115636  005201  MOV     #TTCTB1,5201(R0)
21209
21210  115570  170270  005201  TTC2:   STFPS   @5201(R0)               ;TEST INSTRUCTION.
21211  115574  022701  000000          CMP     #0,R1           ;WAS PC CORRECT AFTER EXECUTION?
21212  115600  001024                  BNE     TTC10           ;BRANCH IF NOT CORRECT.
21213  115602  020027  110445          CMP     R0,#TTCTB2-5201 ;IS R0 CORRECT?
21214  115606  001021                  BNE     TTC10           ;BRANCH IF NOT CORRECT.
21215  115610  023727  115636  103747  CMP     @#TTCTB1,#103747  ;IS THE RESULT CORRECT?
21216  115616  001015                  BNE     TTC10           ;BRANCH IF NOT CORRECT.
21217  115620  023727  115640  177777  CMP     @#TTCTB1+2,#-1  ;IS THE RESULT CORRECT?
21218  115626  001011                  BNE     TTC10           ;BRANCH IF NOT CORRECT.
21219  115630  000411                  BR      TTCDONE
21220
21221                          ;TEST DATA BUFFER:
21222  115632  177777  177777  TTCTB0: .WORD   -1,-1
21223  115636  177777  177777  177777  TTCTB1: .WORD   -1,-1,-1,-1
21224  115644  177777
21225  115646  177777  177777  TTCTB2: .WORD   -1,-1
21226  115652                  TTC10:
21227  115652  104000                  EMT                     ;
21228  115654                  TTCDONE:
21229  115654  004767  002320          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
21230                                                          ;SEE IF THE USER HAS EXPRESSED
21231                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
21232                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21233                                                          ;THE USER TYPED CONTROL G?).
21234
21235                          ;*****************************************************************************
21236                          ;TEST 567       DESTINATION MODES, MODE 2 (FL=1), TEST
21237                          ;*****************************************************************************
21238  115660                  TS567:
21239  115660  012700  000300          MOV     #300,R0         ;SET UP FPS.
21240  115664  170100                  LDFPS   R0
21241  115666  012700  115712          MOV     #UUCTP1,R0      ;SET UP THE ACO OPERAND.
21242  115672  172410                  LDD     (R0),AC0
21243  115674  012700  115724          MOV     #UUCBF0,R0
21244
21245  115700  175420          UUC2:   STCDL   AC0,(R0)+       ;TEST INSTRUCTION.
21246
21247  115702  020027  115730          CMP     R0,#UUCBF0+4    ;IS R0 CORRECT?
```

```
21248  115706  001411                          BEQ     UUCDONE
21249  115710  104000                          EMT                     ;
21250                                  ;TEST DATA BUFFER:
21251  115712  000000  000000  000000  UUCTP1: .WORD   0,0,0,0
21252  115720  000000
21253  115722  177777                          -1
21254  115724  177777  177777  177777  UUCBF0: .WORD   -1,-1,-1
21255
21256  115732                          UUCDONE:
21257  115732  004767  002242                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
21258                                                                  ;SEE IF THE USER HAS EXPRESSED
21259                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
21260                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21261                                                                  ;THE USER TYPED CONTROL G?).
21262
21263                                  ;*****************************************************************************
21264                                  ;TEST 570       DESTINATION MODES, MODE 4 (FL=1), TEST
21265                                  ;*****************************************************************************
21266  115736                          TS570:
21267
21268  115736  012700  000300                  MOV     #300,R0         ;SET UP FPS.
21269  115742  170100                          LDFPS   R0
21270  115744  012700  115770                  MOV     #VVCTP1,R0      ;SET UP THE AC0 OPERAND.
21271  115750  172410                          LDD     (R0),AC0
21272  115752  012700  116006                  MOV     #VVCBF0+4,R0
21273
21274  115756  175440                  VVC2:   STCDL   AC0,-(R0)               ;TEST INSTRUCTION.
21275
21276  115760  020027  116002                  CMP     R0,#VVCBF0      ;IS R0 CORRECT?
21277  115764  001411                          BEQ     VVCDONE
21278  115766  104000                          EMT                     ;
21279                                  ;TEST DATA BUFFER:
21280  115770  000000  000000  000000  VVCTP1: .WORD   0,0,0,0
21281  115776  000000
21282  116000  177777                          -1
21283  116002  177777  177777  177777  VVCBF0: .WORD   -1,-1,-1
21284
21285  116010                          VVCDONE:
21286  116010  004767  002164                  JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
21287                                                                  ;SEE IF THE USER HAS EXPRESSED
21288                                                                  ;THE DESIRE TO CHANGE THE SOFTWARE
21289                                                                  ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21290                                                                  ;THE USER TYPED CONTROL G?).
21291
21292                                  ;*****************************************************************************
21293                                  ;TEST 571       STCDI AND STCDL TEST
21294                                  ;*****************************************************************************
21295  116014                          TS571:
21296
21297                                  ;FIRST TEST STC WITH EXP=100 (EXCESS 200)
21298  116014  004737  116752          WWC1:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21299  116020  020000  000000  000000  1$:     .WORD   20000,0,0,0     ;AC0 OPERAND.
21300  116026  000000
21301  116030  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
21302  116034  177777  177777          3$:     .WORD   -1,-1           ;ERROR RES.
21303  116040  040300                  4$:     40300                   ;FPS BEFORE EXECUTION.
```

```
21304  116042  040304                          40304                   ;FPS AFTER EXECUTION.
21305  116044  140304                          140304                  ;ANTICIPATED ERRONEOUS FPS.
21306  116046  177777                          -1              ;REPORT RESULT INCORRECT.
21307                                  ;EXP=0 (OCT)    FL=1    FIC=0
21308  116050  004737  116752          WWC2:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21309  116054  040000  000000  000000  1$:     .WORD   40000,0,0,0     ;AC     ;AC0 OPERAND.
21310  116062  000000
21311  116064  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
21312  116070  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
21313  116074  040313                  4$:     40313                   ;FPS BEFORE EXECUTION.
21314  116076  040304                          40304                   ;FPS AFTER EXECUTION.
21315  116100  140304                          140304                  ;ANTICIPATED ERRONEOUS FPS.
21316  116102  177777                          -1              ;EXPECTED FEC.
21317                                  ;EXP=37 (OCT)   FL=1    FIC=1
21318  116104  004737  116752          WWC3:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21319  116110  047667  075757  157737  1$:     .WORD   47667,75757,157737,167773       ;AC0 OPERAND.
21320  116116  167773
21321  116120  055675  173757          2$:     .WORD   55675,173757            ;EXPECTED RESULT.
21322  116124  122102  004021          3$:     .WORD   122102,004021           ;ANTICIPATED ERRONEOUS RESULT.
21323  116130  040717                  4$:     40717                   ;FPS BEFORE EXECUTION.
21324  116132  040700                          40700                   ;FPS AFTER EXECUTION.
21325  116134  140705                          140705                  ;ANTICIPATED ERRONEOUS FPS.
21326  116136  177777                          -1              ;EXPECTED FEC.
21327                                  ;EXP=40 (OCT)   FL=1    FIC=1
21328  116140  004737  116752          WWC4:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21329  116144  050000  000000  000000  1$:     .WORD   50000,0,0,0     ;AC0 OPERAND.
21330  116152  000000
21331  116154  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
21332  116160  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
21333  116164  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
21334  116166  140705                          140705                  ;FPS AFTER EXECUTION.
21335  116170  040705                          40705                   ;ANTICIPATED ERRONEOUS FPS.
21336  116172  000006                          6               ;EXPECTED FEC.
21337
21338                                  ;EXP=40 (OCT)    FL=1   FIC=0
21339  116174  004737  116752          WWC5:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21340  116200  050000  000000  000000  1$:     .WORD   50000,0,0,0     ;AC0 OPERAND.
21341  116206  000000
21342  116210  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
21343  116214  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
21344  116220  040312                  4$:     40312                   ;FPS BEFORE EXECUTION.
21345  116222  040305                          40305                   ;FPS AFTER EXECUTION.
21346  116224  140305                          140305                  ;ANTICIPATED ERRONEOUS FPS.
21347  116226  177777                          -1              ;EXPECTED FEC.
21348                                  ;EXP=30 (OCT)   FL=1    FIC=1
21349  116230  004737  116752          WWC6:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21350  116234  046000  000001  000000  1$:     .WORD   46000,1,0,0     ;AC0 OPERAND.
21351  116242  000000
21352  116244  000200  000001          2$:     .WORD   200,1           ;EXPECTED RESULT.
21353  116250  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
21354  116254  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
21355  116256  040700                          40700                   ;FPS AFTER EXECUTION.
21356  116260  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
21357  116262  177777                          -1              ;EXPECTED FEC.
21358                                  ;EXP=27 (OCT)   FL=1    FIC=1
21359  116264  004737  116752          WWC7:   JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
```

```
21360  116270  045600  000001  000000  1$:     .WORD   45600,1,0,0      ;AC0 OPERAND.
21361  116276  000000
21362  116300  000100  000000          2$:     .WORD   100,0            ;EXPECTED RESULT.
21363  116304  177777  177777          3$:     .WORD   -1,-1            ;ANTICIPATED ERRONEOUS RESULT.
21364  116310  040707                  4$:     40707                    ;FPS BEFORE EXECUTION.
21365  116312  040700                          40700                    ;FPS AFTER EXECUTION.
21366  116314  177777                          -1                       ;ANTICIPATED ERRONEOUS FPS.
21367  116316  177777                          -1              ;EXPECTED FEC.
21368                                  ;EXP=17 (OCT)   FL=0    FIC=1
21369  116320  004737  116752          WWC10:  JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
21370  116324  043600  000000  000000  1$:     .WORD   43600,0,0,0      ;AC0 OPERAND.
21371  116332  000000
21372  116334  040000  177777          2$:     .WORD   40000,-1                 ;EXPECTED RESULT.
21373  116340  000000  177777          3$:     .WORD   0,-1             ;ANTICIPATED ERRONEOUS RESULT.
21374  116344  040600                  4$:     40600                    ;FPS BEFORE EXECUTION.
21375  116346  040600                          40600                    ;FPS AFTER EXECUTION.
21376  116350  140604                          140604                   ;ANTICIPATED ERRONEOUS FPS.
21377  116352  177777                          -1              ;EXPECTED FEC.
21378
21379                                   ;EXP=20 (OCT)   FL=0    FIC=1
21380  116354  004737  116752          WWC11:  JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
21381  116360  044000  000000  000000  1$:     .WORD   44000,0,0,0      ;AC0 OPERAND.
21382  116366  000000
21383  116370  000000  177777          2$:     .WORD   0,-1             ;EXPECTED RESULT.
21384  116374  177777  177777          3$:     .WORD   -1,-1            ;ANTICIPATED ERRONEOUS RESULT.
21385  116400  040600                  4$:     40600                    ;FPS BEFORE EXECUTION.
21386  116402  140605                          140605                   ;FPS AFTER EXECUTION.
21387  116404  040600                          40600                    ;ANTICIPATED ERRONEOUS FPS.
21388  116406  000006                          6               ;EXPECTED FEC.
21389                                   ;EXP=10 (OCT), AC NEGATIVE, FL=0, FIC=1
21390  116410  004737  116752          WWC12:  JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
21391  116414  142000  000000  000000  1$:     .WORD   142000,0,0,0     ;AC0 OPERAND.
21392  116422  000000
21393  116424  177600  177777          2$:     .WORD   177600,-1                ;EXPECTED RESULT.
21394  116430  000200  000000          3$:     .WORD   200,0            ;ANTICIPATED ERRONEOUS RESULT.
21395  116434  040600                  4$:     40600                    ;FPS BEFORE EXECUTION.
21396  116436  040610                          40610                    ;FPS AFTER EXECUTION.
21397  116440  040600                          40600                    ;ANTICIPATED ERRONEOUS FPS.
21398  116442  177777                          -1              ;EXPECTED FEC.
21399                                   ;EXP=37 (OCT), FL=1, FIC=1, AC NEG.
21400  116444  004737  116752          WWC13:  JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
21401  116450  147600  000000  000000  1$:     .WORD   147600,0,0,0     ;AC0 OPERAND.
21402  116456  000000
21403  116460  140000  000000          2$:     .WORD   140000,0                 ;EXPECTED RESULT.
21404  116464  137777  000000          3$:     .WORD   137777,0                 ;ANTICIPATED ERRONEOUS RESULT.
21405  116470  040700                  4$:     40700                    ;FPS BEFORE EXECUTION.
21406  116472  040710                          40710                    ;FPS AFTER EXECUTION.
21407  116474  177777                          -1                       ;ANTICIPATED ERRONEOUS FPS.
21408  116476  177777                          -1              ;EXPECTED FEC.
21409                                   ;EXP=37  (OCT),  FL=1,  FIC=1,  AC  NEG
21410  116500  004737  116752          WWC14:  JSR     PC,@#STCSUB      ;GO EXECUTE THE INSTRUCTION.
21411  116504  147600  000000  001000  1$:     .WORD   147600,0,1000,0  ;AC0 OPERAND.
21412  116512  000000
21413  116514  137777  177777          2$:     .WORD   137777,177777            ;EXPECTED RESULT.
21414  116520  140000  177777          3$:     .WORD   140000,177777            ;ANTICIPATED ERRONEOUS RESULT.
21415  116524  040707                  4$:     40707                    ;FPS BEFORE EXECUTION.
```

```
21416  116526  040710                          40710                   ;FPS AFTER EXECUTION.
21417  116530  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
21418  116532  177777                          -1              ;EXPECTED FEC.
21419                                  ;EXP=41 (OCT),  AC NEG, FL=1, FIC=1
21420  116534  004737  116752          WWC15:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21421  116540  150200  000000  000000  1$:     .WORD   150200,0,0,0    ;AC0 OPERAND.
21422  116546  000000
21423  116550  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
21424  116554  177777  177777          3$:     .WORD   -1,-1           ;ANTICIPATED ERRONEOUS RESULT.
21425  116560  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
21426  116562  140705                          140705                  ;FPS AFTER EXECUTION.
21427  116564  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
21428  116566  000006                          6               ;EXPECTED FEC.
21429                                  ;EXP=40 (OCT),  AC NEG,  FL=1,  FIC=1
21430  116570  004737  116752          WWC16:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21431  116574  150000  000001  000000  1$:     .WORD   150000,1,0,0    ;AC0 OPERAND.
21432  116602  000000
21433  116604  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
21434  116610  100000  177600          3$:     .WORD   100000,-200             ;ANTICIPATED ERRONEOUS RESULT.
21435  116614  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
21436  116616  140705                          140705                  ;FPS AFTER EXECUTION.
21437  116620  040700                          40700                   ;ANTICIPATED ERRONEOUS FPS.
21438  116622  000006                          6               ;EXPECTED FEC.
21439                                  ;EXP=40, AC NEGATIVE, FL=1, FIC=1
21440  116624  004737  116752          WWC17:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21441  116630  150001  000000  000000  1$:     .WORD   150001,0,0,0    ;AC0 OPERAND.
21442  116636  000000
21443  116640  000000  000000          2$:     .WORD   0,0             ;EXPECTED RESULT.
21444  116644  077400  000000          3$:     .WORD   77400,0         ;ANTICIPATED ERRONEOUS RESULT.
21445  116650  040700                  4$:     40700                   ;FPS BEFORE EXECUTION.
21446  116652  140705                          140705                  ;FPS AFTER EXECUTION.
21447  116654  177777                          -1                      ;ANTICIPATED ERRONEOUS FPS.
21448  116656  000006                          6               ;EXPECTED FEC.
21449                                  ;EXP 40    (OCT), AC MOST NEG LONG INT, FL=1
21450                                  ;FIC=1
21451  116660  004737  116752          WWC20:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21452  116664  150000  000000  000000  1$:     .WORD   150000,0,0,0    ;AC0 OPERAND.
21453  116672  000000
21454  116674  100000  000000          2$:     .WORD   100000,0                ;EXPECTED RESULT.
21455  116700  000000  000000          3$:     .WORD   0,0             ;ANTICIPATED ERRONEOUS RESULT.
21456  116704  040700                  4$:     40700                           ;FPS BEFORE EXECUTION.
21457  116706  040710                          40710                   ;FPS AFTER EXECUTION.
21458  116710  140705                          140705                  ;ANTICIPATED ERRONEOUS FPS.
21459  116712  177777                          -1              ;EXPECTED FEC.
21460                                  ;EXP=20, AC = MOST NEG INTEGER, FL=0, FIC=1
21461
21462  116714  004737  116752          WWC21:  JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21463  116720  144000  000001  000000  1$:     .WORD   144000,1,0,0    ;AC0 OPERAND.
21464  116726  000000
21465  116730  100000  177777          2$:     .WORD   100000,-1               ;EXPECTED RESULT.
21466  116734  100000  177400          3$:     .WORD   100000,177400           ;ANTICIPATED ERRONEOUS RESULT.
21467  116740  040600                  4$:     40600                   ;FPS BEFORE EXECUTION.
21468  116742  040610                          40610                   ;FPS AFTER EXECUTION.
21469  116744  140605                          140605                  ;ANTICIPATED ERRONEOUS FPS.
21470  116746  177777                          -1              ;EXPECTED FEC.
21471  116750  000457                  6$:     BR      WWCDONE
```

```
21472
21473                   ;THIS SUBROUTINE, STCSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
21474                   ;THE STCDI OR STCDL INSTRUCTION AND CHECK THE RESULTS. A CALL
21475                   ;TO IT IS MADE THUS:
21476                   ;
21477                   ;               JSR     PC,@#STCSUB
21478                   ;ACARG:        .WORD   X,X,X,X         ;AC OPERAND
21479                   ;RES:          .WORD   X,X             ;EXPECTED RESULT
21480                   ;ERRES:        .WORD   X,X             ;ERROR RESULT
21481                   ;FPSB:         .WORD   X               ;FPS BEFORE EXECUTION
21482                   ;FPSA:         .WORD   X               ;FPS AFTER EXECUTION
21483                   ;ERFPS:        .WORD   X               ;ERROR FPS.
21484                   ;FEC:          .WORD   X               ;EXPECTED FEC
21485                   ;ERR1:         ERROR   X               ;DATA ERROR.
21486                   ;              BR      CONT
21487                   ;ERR2:         ERROR   X               ;FPS ERROR.
21488                   ;CONT:                                 ;RETURN ADDRESS
21489                   ;
21490                   ;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
21491                   ;THE STCDI OR STCDL INSTRUCTION IS EXECUTED.
21492                   ;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
21493                   ;COMPARED WITH FPSA IF THIS TOO IS CORRECT STCSUB RETURNS CONTROL
21494                   ;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STCSUB
21495                   ;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STCSUB WILL RETURN
21496                   ;TO THE ERROR CALL AT ERR2, OTHERWISE STCSUB ITSELF
21497                   ;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
21498                   ;STCDI OR STCDL IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
21499                   ;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
21500                   ;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STCSUB
21501                   ;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
21502                   ;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STCSUB WILL
21503                   ;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
21504
21505  116752  012601        STCSUB: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
21506  116754  012700  000200        MOV     #200,R0         ;SET UP THE AC0 OPERAND.
21507  116760  170100                LDFPS   R0
21508  116762  010100                MOV     R1,R0
21509  116764  172410                LDD     (R0),AC0
21510  116766  012702  117100        MOV     #STCIBF,R2      ;INITIALIZE THE OUT PUT BUFFER.
21511  116772  012700  000004        MOV     #4,R0
21512  116776  012722  177777  1$:   MOV     #-1,(R2)+
21513  117002  077003                SOB     R0,1$
21514  117004  016100  000020        MOV     20(R1),R0       ;SET THE FPS.
21515  117010  170100                LDFPS   R0
21516  117012  012700  117100        MOV     #STCIBF,R0
21517  117016  175410          2$:   STCDL   AC0,(R0)        ;TEST INSTRUCTION.
21518
21519  117020  170204                STFPS   R4              ;GET THE FPS.
21520  117022  170305                STST    R5              ;GET THE FEC.
21521  117024  010102                MOV     R1,R2
21522  117026  062702  000010        ADD     #10,R2
21523  117032  012700  117100        MOV     #STCIBF,R0      ;SEE IF THE RESULT IS CORRECT.
21524  117036  012703  000002        MOV     #2,R3
21525  117042  022022          3$:   CMP     (R0)+,(R2)+
21526  117044  001014                BNE     10$
21527  117046  077303                SOB     R3,3$
```

```
21528  117050  016102  000022          MOV     22(R1),R2
21529  117054  020204                  CMP     R2,R4           ;SEE IF THE FPS IS CORRECT.
21530  117056  001007                  BNE     10$             ;BRANCH IF INCORRECT.
21531  117060  005702                  TST     R2
21532  117062  100003                  BPL     4$
21533  117064  026105  000026          CMP     26(R1),R5       ;SEE IF THE FEC IS CORRECT.
21534  117070  001002                  BNE     10$             ;BRANCH IF INCORRECT.
21535
21536  117072  000161  000030  4$:     JMP     30(R1)          ;RETURN.
21537  117076                  10$:
21538  117076  104000                  EMT                     ;
21539
21540                          ;DATA BUFFER:
21541  117100  177777  177777  177777 STCIBF: .WORD  -1,-1,-1,-1
21542  117106  177777
21543
21544  117110                  WWCDONE:
21545  117110  004767  001064          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
21546                                                          ;SEE IF THE USER HAS EXPRESSED
21547                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
21548                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21549                                                          ;THE USER TYPED CONTROL G?).
21550
21551
21552                          ;*****************************************************************************
21553                          ;TEST 572       STCFL AND STCFI TEST
21554                          ;*****************************************************************************
21555  117114                  TS572:
21556
21557
21558                          ;EXPONENT=37, FL=1
21559  117114  004737  116752          JSR     PC,@#STCSUB     ;GO EXECUTE THE INSTRUCTION.
21560  117120  047777  177777  177777 1$:     .WORD   47777,-1,-1,-1  ;AC0 OPERAND.
21561  117126  177777
21562  117130  077777  177600  2$:     .WORD   77777,177600            ;EXPECTED RESULT.
21563  117134  077777  177777  3$:     .WORD   77777,177777            ;ANTICIPATED ERRONEOUS RESULT.
21564  117140  040100          4$:     40100                   ;FPS BEFORE EXECUTION.
21565  117142  040100                  40100                   ;FPS AFTER EXECUTION.
21566  117144  177777                  -1                      ;ANTICIPATED ERRONEOUS FPS.
21567  117146  177777                  -1              ;EXPECTED FEC.
21568  117150                  XXCDONE:
21569  117150  004767  001024          JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
21570                                                          ;SEE IF THE USER HAS EXPRESSED
21571                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
21572                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21573                                                          ;THE USER TYPED CONTROL G?).
21574
21575
21576                          ;*****************************************************************************
21577                          ;TEST 573       STEXP TEST
21578                          ;*****************************************************************************
21579  117154                  TS573:
21580
21581                          ; EXP = 100 (EXCESS 200)
21582  117154  004737  117362  YYC1:   JSR     PC,@#STXSUB
21583  117160  020000  000000  000000 1$:     .WORD   20000,0,0,0                     ;AC
```

```
21584  117166  000000                  2$:     -100                            ;EXP RES
21585  117170  177700                  3$:     52525                           ;ERROR EXP.
21586  117172  052525                  4$:     40000                           ;FPSB
21587  117174  040000                          40010                           ;FPSA
21588  117176  040010                          40000                           ;ERROR FPS
21589  117200  040000
21590                                  ; EXP = 200 (EXCESS 200)
21591  117202  004737  117362          YYC2:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
21592  117206  040000  000000  000000  1$:     .WORD   40000,0,0,0     ;AC0 OPERAND.
21593  117214  000000
21594  117216  000000                  2$:     0                               ;EXPECTED EXPONENT RESULT.
21595  117220  052525                  3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
21596  117222  040000                  4$:     40000                           ;FPS BEFORE EXECUTION.
21597  117224  040004                          40004                           ;FPS AFTER EXECUTION.
21598  117226  040000                          40000                           ;ANTICIPATED ERRONEOUS FPS.
21599                                  ; EXP = 201 (EXCESS 200)
21600  117230  004737  117362          YYC3:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
21601  117234  040200  000000  000000  1$:     .WORD   40200,0,0,0     ;AC0 OPERAND.
21602  117242  000000
21603  117244  000001                  2$:     1                               ;EXPECTED EXPONENT RESULT.
21604  117246  052525                  3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
21605  117250  040000                  4$:     40000                           ;FPS BEFORE EXECUTION.
21606  117252  040000                          40000                           ;FPS AFTER EXECUTION.
21607  117254  040004                          40004                           ;ANTICIPATED ERRONEOUS FPS.
21608                                  ; EXP = 375  (EXCESS 200)
21609
21610  117256  004737  117362          YYC4:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
21611  117262  077200  000000  000000  1$:     .WORD   77200,0,0,0     ;AC0 OPERAND.
21612  117270  000000
21613  117272  000175                  2$:     175                             ;EXPECTED EXPONENT RESULT.
21614  117274  052525                  3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
21615  117276  040000                  4$:     40000                           ;FPS BEFORE EXECUTION.
21616  117300  040000                          40000                           ;FPS AFTER EXECUTION.
21617  117302  040010                          40010                           ;ANTICIPATED ERRONEOUS FPS.
21618                                  ; EXP = 1  (EXCESS 200)
21619
21620  117304  004737  117362          YYC5:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
21621  117310  000200  000000  000000  1$:     .WORD   200,0,0,0       ;AC0 OPERAND.
21622  117316  000000
21623  117320  177601                  2$:     -177                            ;EXPECTED EXPONENT RESULT.
21624  117322  052525                  3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
21625  117324  040000                  4$:     40000                           ;FPS BEFORE EXECUTION.
21626  117326  040010                          40010                           ;FPS AFTER EXECUTION.
21627  117330  040000                          40000                           ;ANTICIPATED ERRONEOUS FPS.
21628                                  ; EXP = 156  (EXCESS 200)
21629
21630  117332  004737  117362          YYC6:   JSR     PC,@#STXSUB     ;GO EXECUTE THE INSTRUCTION.
21631  117336  033400  000000  000000  1$:     .WORD   33400,0,0,0     ;AC0 OPERAND.
21632  117344  000000
21633  117346  177756                  2$:     -22                             ;EXPECTED EXPONENT RESULT.
21634  117350  052525                  3$:     52525           ;ANTICIPATED ERRONEOUS RESULT.
21635  117352  047707                  4$:     47707                           ;FPS BEFORE EXECUTION.
21636  117354  047710                          47710                           ;FPS AFTER EXECUTION.
21637  117356  177777                          -1                              ;ANTICIPATED ERRONEOUS FPS.
21638  117360  000452                          BR      YYCDONE
21639
```

```
21640
21641
21642
21643
21644
21645
21646
21647
21648
21649
21650
21651
21652
21653
21654
21655
21656
21657
21658
21659
21660
21661
21662
21663
21664
21665
21666
21667
21668
21669
21670
21671  117362  012601                        STXSUB: MOV     (SP)+,R1        ;GET A POINTER TO THE ARGUMENTS.
21672  117364  010102                                MOV     R1,R2
21673  117366  012737  123456  117474                MOV     #123456,@#STXBF
21674  117374  012737  076543  117476                MOV     #76543,@#STXBF+2
21675  117402  012700  000200                        MOV     #200,R0
21676  117406  170100                                LDFPS   R0
21677  117410  010100                                MOV     R1,R0           ;SET UP THE AC0 OPERAND.
21678  117412  172410                                LDD     (R0),AC0
21679  117414  016100  000016                        MOV     16(R1),R0       ;SET THE FPS.
21680  117420  170100                                LDFPS   R0
21681  117422  012700  117474                        MOV     #STXBF,R0
21682  117426  175010                          1$:   STEXP   AC0,(R0)        ;TEST INSTRUCTION.
21683  117430  170204                                STFPS   R4              ;GET FPS.
21684  117432  026137  000010  117474                CMP     10(R1),@#STXBF  ;WAS RESULT CORRECT?
21685  117440  001401                                BEQ     5$
21686  117442  104000                                EMT                     ;
21687  117444  020461  000016                  5$:   CMP     R4,16(R1)       ;SEE IF THE FPS IS CORRECT.
21688  117450  001401                                BEQ     10$
21689  117452  104000                                EMT                     ;
21690                                  ;SEE IF MORE THAN ONE WORD WAS WRITTEN IN THE OUTPUT BUFFER.
21691  117454  022737  076543  117476  10$:   CMP     #76543,@#STXBF+2
21692  117462  001401                                BEQ     4$
21693  117464  104000                                EMT                     ;
21694  117466  000161  000022                  4$:   JMP     22(R1)
21695
```

```
;THIS SUBROUTINE, STXSUB, IS USED TO SET UP THE OPERANDS, EXECUTE
;THE STEXP INSTRUCTION AND CHECK THE RESULTS. A CALL
;TO IT IS MADE THUS:
;
;               JSR     PC,@#STXSUB
;               ACARG:  .WORD   X,X,X,X         ;AC OPERAND
;               RES:    .WORD   X               ;EXPECTED RESULT
;               ERRES:  .WORD   X               ;ERROR RESULT
;               FPSB:   .WORD   X               ;FPS BEFORE EXECUTION
;               FPSA:   .WORD   X               ;FPS AFTER EXECUTION
;               ERFPS:  .WORD   X               ;ERROR FPS.
;               ERR1:   ERROR   X               ;DATA ERROR.
;                       BR      CONT
;               ERR2:   ERROR   X               ;FPS ERROR.
;               CONT:                           ;RETURN ADDRESS
;
;THE OPERANDS ARE SET UP (USING AC0 AS THE ACCUMULATOR). THEN
;THE STEXP INSTRUCTION IS EXECUTED.
;THE RESULT IS CHECKED AGAINST RES. IF THE RESULT IS CORRECT THEN THE FPS IS
;COMPARED WITH FPSA IF THIS TOO IS CORRECT STXSUB RETURNS CONTROL
;TO THE CALLING ROUTINE AT CONT. IF THE FPS IS BAD STXSUB
;COMPARE IT TO ERROR FPS. IF THIS MATCHES THEN STXSUB WILL RETURN
;TO THE ERROR CALL AT ERR2, OTHERWISE STXSUB ITSELF
;REPORTS THIS FAILURE AND THEN RETURNS TO CONT. IF THE RESULT OF THE
;STEXP IS INCORRECT, THE INCORRECT RESULT IS COMPARED WITH THE
;ANTICIPATED FAILING DATA PATTERN, ERRES. IF THE FAILURE IN
;THE RESULT WAS ANTICIPATED CORRECTLY TO BE ERRES THEN STXSUB
;WILL TRANSFER CONTROL TO THE ERROR CALL AT ERR1. OTHERWISE THE
;RESULT WAS INCORRECT BUT WAS NOT ANTICIPATED AND STXSUB WILL
;REPORT THE FAILURE AFTER WHICH CONTROL WILL BE PASSED TO CONT.
```

```
                                        -1
117472  177777
117474  177777  177777  177777  STXBF:  .WORD   -1,-1,-1,-1,-1
117502  177777  177777
                                YYCDONE:
117506
117506  004767  000466                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
                                ;*****************************************************************************
                                ;TEST 574       STST TEST
                                ;*****************************************************************************
117512                          TS574:
                                        MOV     #40000,R0       ;SET FPS. FID=1.
117512  012700  040000
117516  170100                        LDFPS   R0
                                ZZC2:   .WORD   170003                  ;ILLEGAL FPP
117520  170003                                                         ;OP CODE
                                        MOV     #ZZCBF,R0       ;SET UP THE OUTPUT BUFFER.
117522  012700  117602
117526  012710  177777                MOV     #-1,(R0)
117532  012760  177777  000002        MOV     #-1,2(R0)
                                ZZC3:   STST    (R0)                    ;GET FEC AND
117540  170310                                                           ;FEA
                                        STFPS   R4              ;GET FPS.
117542  170204
117544  012700  117602                MOV     #ZZCBF,R0
117550  022710  000002                CMP     #2,(R0) ;SEE IF FEC IS CORRECT.
117554  001010                        BNE     ZZC10           ;BRANCH IF INCORRECT.
117556  022760  117520  000002        CMP     #ZZC2,2(R0)     ;SEE IF FEA, ADDRESS, IS CORRECT.
117564  001004                        BNE     ZZC10           ;BRANCH IF INCORRECT.
117566  022704  140000                CMP     #140000,R4      ;SEE IF FPS IS CORRECT.
117572  001001                        BNE     ZZC10           ;BRANCH IF INCORRECT.
117574  000407                        BR      ZZCDONE
117576                          ZZC10:
117576  104000                        EMT                     ;
                                ;DATA BUFFER:
                                        -1
117600  177777
117602  177777  177777  177777  ZZCBF:  .WORD   -1,-1,-1,-1
117610  177777
117612  177777                          -1
                                ZZCDONE:
117614
117614  004767  000360                JSR     PC,..RSET       ;GO INITIALIZE THE FPS AND STACK; AND
                                                                ;SEE IF THE USER HAS EXPRESSED
                                                                ;THE DESIRE TO CHANGE THE SOFTWARE
                                                                ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
                                                                ;THE USER TYPED CONTROL G?).
                                ;*****************************************************************************
                                ;TEST 575       SPECIAL CASE TEST
                                ;*****************************************************************************
117620                          TS575:
```

```
21752  117620  012746  144724          AAD1:   MOV     #144724, -(SP)  ;PUT FRACTION ON STACK
21753  117624  012746  040600                  MOV     #40600, -(SP)   ;PUT EXPONENT ON STACK
21754  117630  005046                          CLR     -(SP)           ;PUT SUBTRAHEND FRACTION ON STACK
21755  117632  012746  040600                  MOV     #40600, -(SP)   ;PUT SUBTRAHEND EXPONENT ON STACK
21756  117636  172466  000004                  LDF     4(SP), AC0      ;LOAD FP ACCUMULATORS
21757  117642  173026                          SUBF    (SP)+, AC0      ;DO SUBTRACTION
21758  117644  174037  117674                  STF     AC0, @#AADBF    ;GET AND STORE ANSWER
21759  117650  022737  036711  117674          CMP     #36711, @#AADBF ;IS EXPONENT CORRECT
21760  117656  001401                          BEQ     1$
21761  117660  104000                          EMT                     ;BAD EXPONENT FROM SUBTRACTION
21762  117662  022737  152000  117676  1$:     CMP     #152000, @#AADBF+2 ;IS FRACTION CORRECT
21763  117670  001403                          BEQ     AADDONE
21764  117672  104000                          EMT                     ;FRACTION INCORRECT
21765
21766  117674  000000          AADBF:  .WORD   0
21767  117676  000000                  .WORD   0
21768
21769  117700  012706  001000  AADDONE: MOV     #STBOT, SP      ;RESTORE STACK POINTER
21770  117704  004767  000270          JSR     PC,.RSET        ;GO INITIALIZE THE FPS AND STACK; AND
21771                                                          ;SEE IF THE USER HAS EXPRESSED
21772                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
21773                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21774                                                          ;THE USER TYPED CONTROL G?).
21775
21776
21777
21778                          ;*****************************************************************************
21779                          ;TEST 576       INTERRUPTABILITY TEST
21780                          ;*****************************************************************************
21781  117710          TS576:
21782  117710  004567  130352  BBD1:   JSR     R5,CHKAPT
21783  117714  000520          BR      FPEXIT          ;SKIP TEST IF ON APT AND NOT FIRST PASS
21784
21785  117716  005001          CLR     R1              ;INITIALIZE A COUPLE OF COUNTERS
21786  117720  005000          CLR     R0
21787  117722  013767  000064  117310  MOV     @#64,   $TMP0   ;SAVE INTERRUPT VECTOR
21788  117730  013767  000066  117304  MOV     @#66,   $TMP1   ;SAVE INTERRUPT PRIORITY
21789  117736  012737  120010  000064  MOV     #3$,    @#64    ;SET UP INTERRUPT PRIORITY FOR THIS TEST
21790  117744  005037  000066          CLR     @#66            ;AND PRIORITY
21791  117750  005067  060022          CLR     PS              ;PUT PROCESSOR PRIORITY AT 0
21792  117754  005067  057606          CLR     TPB             ;SEND A NULL CHARACTER
21793  117760  105767  057600  1$:     TSTB    TTCSR           ;WAIT FOR DONE TO SET
21794  117764  100375          BPL     1$
21795  117766  005067  057574          CLR     TPB             ;SEND A SECOND CHARACTER
21796  117772  052767  000100  057564  BIS     #BIT6,  TTCSR   ;SET INTERRUPT ENABLE
21797  120000  005200  2$:     INC     R0              ;INCREMENT COUNTER TO GET BASE TIME
21798  120002  001376          BNE     2$              ;CONTINUE LOOPING UNLESS COUNTER GOES TO 0
21799  120004  000005          RESET                   ;IF NO INTERRUPT YET KILL IT
21800  120006  104000          EMT                     ;NO INTERRUPT OCCURRED IN ALLOTTED TIME
21801  120010  166700  000110  3$:     SUB     Y,      R0      ;SUBTRACT TIME FOR FP INSTRUCTION
21802  120014  010067  000106          MOV     R0,     Z       ;SAVE FIRST TIME
21803  120020  012737  120076  000064  MOV     #7$,    @#64    ;SET UP FOR NEXT INTERRUPT
21804  120026  005100  4$:     COM     R0              ;MAKE PRE LOOP COUNTER NEGATIVE
21805  120030  005067  057530          CLR     TTCSR           ;MAKE SURE NO INTERRUPT YET
21806  120034  005067  057526          CLR     TPB             ;SEND A CHARACTER
21807  120040  105767  057520  5$:     TSTB    TTCSR           ;WAIT FOR READY BIT TO SET
```

```
21808  120044  100375                        BPL     5$
21809  120046  005067  057514                CLR     TPB                     ;SEND SECOND CHARACTER
21810  120052  052767  000100  057504        BIS     #BIT6,  TTCSR           ;SET INTERRUPT ENABLE
21811  120060  00520C                6$:     INC     R0                      ;DO PRE LOOP
21812  120062  001?                          BNE     6$
21813  120064  171227  040400                MULF    #2.,    AC2             ;DO FLOATING POINT INSTRUCTION
21814  120070  000240                        NOP                             ;JUST IN CASE INTERRUPT TAKES TOO LONG
21815  120072  000005                        RESET                           ;IF NO INTERRUPT CLEAR THE WORLD
21816  120074  104000                        EMT                             ;INTERRUPT NOT BACK IN ALLOTTED TIME
21817  120076  005201                7$:     INC     R1                      ;INCREMENT TIMES THROUGH COUNTER
21818  120100  020127  000015                CMP     R1,     #15             ;HAVE WE PASSED HERE 15 TIMES BEFORE
21819  120104  001411                        BEQ     BBDDONE                 ;IF YES I MAY NEVER PASS HERE AGAIN
21820  120106  062767  000002  000012        ADD     #2.,    Z               ;IF NO ADD A LITTLE TIME TO PRELOOP
21821  120114  016700  000006                MOV     Z,      R0              ;PUT NEW COUNT IN COUNTER
21822  120120  000742                        BR      4$                      ;DO IT ALL AGAIN
21823
21824  120122  000000                X:      .WORD   0
21825  120124  000026                Y:      .WORD   26
21826  120126  000000                Z:      .WORD   0
21827
21828  120130  042767  000100  057426  BBDDONE:BIC     #100,   TTCSR           ;CLEAR INTERRUPT ENABLE BEFORE EXITING TEST
21829  120136  016737  117076  000064          MOV     $TMP0,  @#64            ;RESTORE PRINTER VECTOR
21830  120144  016737  117072  000066          MOV     $TMP1,  @#66            ;RESTORE PRINTER PRIORITY
21831  120152  004767  000022                  JSR     PC,..RSET               ;GO INITIALIZE THE FPS AND STACK; AND
21832                                                                          ;SEE IF THE USER HAS EXPRESSED
21833                                                                          ;THE DESIRE TO CHANGE THE SOFTWARE
21834                                                                          ;VIRTUAL CONSOLE SWITCH REGISTER (HAS
21835                                                                          ;THE USER TYPED CONTROL G?).
21836  120156  000167  000102          FPEXIT: JMP     SLU1ST                  ;GET OVER SUBROUTINES TO NEXT TEST
21837
21838
21839  120162  012737  000003  001002  ERROR4: MOV     #3,@#$FATAL                     ;SET UP FATAL ERROR NUMBER
21840  120170  012767  000001  060602          MOV     #1,$MSGTY                       ;SET FATAL ERROR FLAG
21841  120176  000777                  FPHLT:  BR      .                       ;STAY HERE FOREVER
21842                                          .SBTTL  FLAG RESET ROUTINE
21843                                  ;*******************************************************************************
21844                                  ;*THIS ROUTINE WILL BE CALLED AT THE END OF EACH FLOATING POINT TEST
21845                                  ;*TO RESET THE STACK, CLEAR THE FPS AND REINITIALIZE TRAP VECTORS
21846
21847  120200  012737  120162  000244  .RSET:   MOV     #ERROR4,@#FPVECT
21848  120206  012737  021216  000004          MOV     #T04,@#ERRVECT
21849  120214  012737  021220  000010          MOV     #T010,@#10
21850  120222  011600                          MOV     (SP),R0
21851  120224  012706  001000                  MOV     #STBOT,SP
21852  120230  005004                          CLR     R4
21853  120232  170104                          LDFPS   R4
21854  120234  000110                          JMP     (R0)
21855
21856                                  ;THESE ARE SOME EQUATES USE IN THE LAST THREE SECTIONS
21857          000001                          BIT0=000001
21858          000002                          BIT1=000002
21859          000004                          BIT2=000004
21860          000010                          BIT3=000010
21861          000020                          BIT4=000020
21862          000040                          BIT5=000040
21863          000100                          BIT6=000100
```

```
21864                 000200                          BIT7=000200
21865                 000400                          BIT8=000400
21866                 001000                          BIT9=001000
21867                 002000                          BIT10=002000
21868                 004000                          BIT11=004000
21869                 010000                          BIT12=010000
21870                 020000                          BIT13=020000
21871                 040000                          BIT14=040000
21872                 100000                          BIT15=100000
21873
21874  120236  177560                         RCSR:   177560                  ;ADDRESS OF RECEIVER COMMAND/STATUS REGISTER
21875  120240  177562                         RBUF:   177562                  ;ADDRESS OF RECEIVER BUFFER
21876  120242  177564                         TCSR:   177564                  ;ADDRESS OF TRANSMITTER COMMAND/STATUS REGISTER
21877  120244  177566                         TBUF:   177566                  ;ADDRESS OF TRANSMITTER BUFFER
21878  120246  000060                         RVECT:  60                      ;RECEIVER INTERRUPT VECTOR
21879  120250  000062                         RPSW:   62
21880  120252  000064                         TVECT:  64                      ;TRANSMITTER INTERRUPT VECTOR
21881  120254  000066                         TPSW:   66
21882
21883                                         ;REAL TIME CLOCK REGISTER AND VECTOR ADDRESSES
21884  120256  177546                         LKS:    .WORD   177546
21885  120260  000100                         RTCVT:  .WORD   100
21886  120262  000102                         RTCPSW: .WORD   102
21887
21888  120264  032777  000004  100772  SLU1ST: BIT     #4,@SWR
21889  120272  001402                          BEQ     1$
21890  120274  000167  002372                  JMP     KWSTRT
21891  120300  012737  000004  001004  1$:     MOV     #4,@#$TESTN     ;PUT TEST NUMBER IN MAILBOX
21892  120306  012737  122654  000030          MOV     #ERROR5,@#30    ;SET UP FOR CORRECT ERROR CALL
21893
21894
21895                                         ;*****************************************************************************
21896                                         ;TEST 577       TEST ABILITY TO REFERENCE TCSR
21897                                         ;*****************************************************************************
21898  120314                                 TS577:
21899  120314  013703  000004                          MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
21900  120320  012737  120334  000004                  MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
21901  120326  005777  177710                          TST     @TCSR           ;REFERENCE THE XMIT COMMAND/STATUS REG.
21902  120332  000401                                  BR      4$
21903  120334                                 1$:
21904  120334  104000                                  EMT                     ;
21905  120336  010337  000004                  4$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
21906
21907
21908
21909                                         ;*****************************************************************************
21910                                         ;TEST 600       TEST ABILITY TO REFERENCE TBUF
21911                                         ;*****************************************************************************
21912  120342                                 TS600:
21913  120342  013703  000004                          MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
21914  120346  012737  120362  000004                  MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
21915  120354  005777  177664                          TST     @TBUF           ;REFERENCE THE XMIT BUFFER
21916  120360  000401                                  BR      4$
21917  120362                                 1$:
21918  120362  104000                                  EMT                     ;
21919  120364  010337  000004                  4$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
```

```
21920
21921
21922                                   ;*****************************************************************************
21923                                   ;TEST 601       TEST THAT TCSR BIT7(DONE) CLEARS WHEN XBUF IS LOADED
21924                                   ;*****************************************************************************
21925  120370                           TS601:
21926  120370  032737  000001  001020           BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
21927  120376  001405                           BEQ     70$             ;IF NO THEN SERIES OF TESTS
21928  120400  005737  001006                   TST     @#$PASS         ;IS THIS FIRST PASS
21929  120404  001402                           BEQ     70$             ;IF YES THEN DO SERIES OF TESTS
21930  120406  000167  002260                   JMP     KWSTRT          ;IF NO THEN BYPASS SERIES OF TESTS
21931  120412  005077  177626           70$:    CLR     @TBUF           ;LOAD XBUF
21932  120416  105777  177620                   TSTB    @TCSR           ;CHECK DONE
21933  120422  100006                           BPL     3$              ;BR IF CLEAR
21934                                                                   ;FILL SECOND BUFFER BECUASE REFRESH COULD CAUSE
21935                                                                   ;FIRST TEST TO FAIL
21936  120424  005077  177614                   CLR     @TBUF           ;FILL DOUBLE BUFFER
21937  120430  105777  177606                   TSTB    @TCSR           ;CHECK DONE
21938  120434  100001                           BPL     3$
21939  120436  104000                           EMT                     ;
21940  120440  005000                   3$:     CLR     R0              ;CLEAR TIMER
21941  120442  105777  177574           4$:     TSTB    @TCSR           ;CHECK FOR XMIT DONE
21942  120446  100403                           BMI     5$              ;IF DONE SETS, BR TO END OF TEST
21943  120450  005200                           INC     R0              ;INCREMENT TIMER
21944  120452  001373                           BNE     4$
21945  120454  104000                           EMT                     ;
21946  120456                           5$:
21947
21948
21949                                   ;*****************************************************************************
21950                                   ;TEST 602       TEST THAT TCSR 'DONE'' SETS WITH RESET
21951                                   ;*****************************************************************************
21952  120456                           TS602:
21953  120456  005077  177562                   CLR     @TBUF           ;LOAD TRANSMIT BUFFER
21954  120462  105777  177554           1$:     TSTB    @TCSR           ;WAIT FOR DONE
21955  120466  100375                           BPL     1$
21956  120470  005077  177550                   CLR     @TBUF           ;LOAD SECOND BUFFER
21957  120474  000240                           NOP
21958  120476  000005                           RESET                   ;SET DONE WITH RESET
21959  120500  105777  177536                   TSTB    @TCSR           ;CHECK FOR DONE SET
21960  120504  100401                           BMI     TS603
21961  120506  104000                           EMT                     ;
21962
21963
21964
21965                                   ;*****************************************************************************
21966                                   ;TEST 603       TEST ABILITY TO ACCESS RCSR
21967                                   ;*****************************************************************************
21968  120510                           TS603:
21969  120510  013703  000004                   MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
21970  120514  012737  120530  000004           MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
21971  120522  005777  177510                   TST     @RCSR           ;ACCESS RCSR
21972  120526  000401                           BR      2$
21973  120530                           1$:
21974  120530  104000                           EMT                     ;
21975  120532  010337  000004           2$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
```

```
                                ;*****************************************************************************
                                ;TEST 604       TEST ABILITY TO ACCESS RBUF
                                ;*****************************************************************************
                                ;
120536                          TS604:
120536 013703 000004                    MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
120542 012737 120556 000004             MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
120550 005777 177464                    TST     @RBUF           ;ACCESS RBUF
120554 000401                           BR      2$
120556                          1$:
120556 104000                           EMT                     ;
120560 010337 000004            2$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;*****************************************************************************
                                ;TEST 605       TEST THAT BIT2(MAINT. BIT) CAN BE SET & RESET
                                ;*****************************************************************************
120564                          TS605:
120564 032777 000004 177450             BIT     #BIT2,@TCSR     ;TEST FOR BIT2 OF TCSR CLEAR
120572 001401                           BEQ     3$
120574 104000                           EMT                     ;
120576 052777 000004 177436     3$:     BIS     #BIT2,@TCSR     ;SET BIT2 OF TCSR
120604 032777 000004 177430             BIT     #BIT2,@TCSR     ;TEST FOR BIT2 SET
120612 001001                           BNE     4$
120614 104000                           EMT                     ;
120616 042777 000004 177416     4$:     BIC     #BIT2,@TCSR     ;CLEAR BIT2 OF TCSR
120624 032777 000004 177410             BIT     #BIT2,@TCSR     ;TEST BIT2 CLEAR
120632 001401                           BEQ     7$
120634 104000                           EMT                     ;
120636 052777 000004 177376     7$:     BIS     #BIT2,@TCSR     ;SET BIT2 OF TCSR
120644 000005                           RESET                   ;CLEAR BIT2 WITH RESET
120646 032777 000004 177366             BIT     #BIT2,@TCSR     ;TEST FOR BIT2 CLEAR
120654 001401                           BEQ     TS606
120656 104000                           EMT                     ;
                                ;
                                ;
                                ;*****************************************************************************
                                ;TEST 606       TEST THAT BIT6(XMIT INT EN) CAN BE SET & RESET
                                ;*****************************************************************************
120660                          TS606:
120660 017703 177366                    MOV     @TVECT,R3       ;SAVE XMIT VECTOR
120664 012777 120706 177360             MOV     #1$,@TVECT      ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
120672 106427 000340                    MTPS    #340            ;SET PSW TO PRIORITY 7
120676 032777 000100 177336             BIT     #BIT6,@TCSR     ;TEST BIT6 OF TCSR
120704 001401                           BEQ     2$
120706                          1$:
120706 104000                           EMT                     ;
120710 052777 000100 177324     2$:     BIS     #BIT6,@TCSR     ;SET BIT6 OF TCSR
120716 032777 000100 177316             BIT     #BIT6,@TCSR     ;TEST BIT6 OF TCSR
120724 001001                           BNE     3$
120726 104000                           EMT                     ;
```

```
22032  120730  042777  000100  177304  3$:     BIC     #BIT6,@TCSR     ;CLEAR BIT6 OF TCSR
22033  120736  032777  000100  177276          BIT     #BIT6,@TCSR     ;TEST BIT6 OF TCSR
22034  120744  001401                          BEQ     4$
22035  120746  104000                          EMT                     ;
22036  120750                          4$:
22037  120750  052777  000100  177264          BIS     #BIT6,@TCSR     ;SET BIT6 OF TCSR
22038  120756  000005                          RESET                   ;CLEAR BIT6 WITH RESET
22039  120760  032777  000100  177254          BIT     #BIT6,@TCSR     ;TEST BIT6 OF TCSR
22040  120766  001401                          BEQ     5$
22041  120770  104000                          EMT                     ;
22042  120772  010377  177254          5$:     MOV     R3,@TVECT       ;RESTORE XMIT VECTOR
22043
22044
22045                                  ;*********************************************************************************
22046                                  ;TEST 607       TEST THAT BIT6 OF RCSR CAN BE SET & RESET
22047                                  ;*********************************************************************************
22048  120776                          TS607:
22049  120776  017703  177244                  MOV     @RVECT,R3       ;SAVE RECEIVE VECTOR
22050  121002  012777  121024  177236          MOV     #1$,@RVECT      ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
22051  121010  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22052  121014  032777  000100  177214          BIT     #BIT6,@RCSR     ;TEST BIT6 OF RCSR
22053  121022  001401                          BEQ     2$
22054  121024                          1$:
22055  121024  104000                          EMT                     ;
22056  121026  052777  000100  177202  2$:     BIS     #BIT6,@RCSR     ;SET BIT6 OF RCSR
22057  121034  032777  000100  177174          BIT     #BIT6,@RCSR     ;TEST BIT6 OF RCSR
22058  121042  001001                          BNE     3$
22059  121044  104000                          EMT                     ;
22060  121046  042777  000100  177162  3$:     BIC     #BIT6,@RCSR     ;CLEAR BIT6 OF RCSR
22061  121054  032777  000100  177154          BIT     #BIT6,@RCSR     ;TEST BIT6 OF RCSR
22062  121062  001401                          BEQ     4$
22063  121064  104000                          EMT                     ;
22064  121066                          4$:
22065  121066  052777  000100  177142          BIS     #BIT6,@RCSR     ;SET BIT6 OF RCSR
22066  121074  000005                          RESET                   ;CLEAR BIT6OF RCSR WITH RESET
22067  121076  032777  000100  177132          BIT     #BIT6,@RCSR     ;TEST BIT6 OF RCSR
22068  121104  001401                          BEQ     5$
22069  121106  104000                          EMT                     ;
22070  121110  010377  177132          5$:     MOV     R3,@RVECT       ;RESTORE RECEIVE VECTOR
22071
22072
22073
22074                                  ;*********************************************************************************
22075                                  ;TEST 610       TEST THAT XMIT INTERRUPTS ONLY WHEN ENABLED
22076                                  ;*********************************************************************************
22077  121114                          TS610:
22078  121114  042777  000100  177120          BIC     #BIT6,@TCSR     ;CLEAR TRANSMIT INTERRUPT ENABLE
22079  121122  017703  177124                  MOV     @TVECT,R3       ;SAVE XMIT VECTOR
22080  121126  012777  121150  177116          MOV     #2$,@TVECT      ;POINT XMIT VECTOR TO ERROR REPORT
22081  121134  105777  177102          1$:     TSTB    @TCSR           ;WAIT FOR DONE
22082  121140  100375                          BPL     1$
22083  121142  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22084  121146  000401                          BR      3$
22085  121150                          2$:
22086  121150  104000                          EMT                     ;
22087  121152  012777  121172  177072  3$:     MOV     #4$,@TVECT      ;SET XMIT VECTOR TO END OF TEST
```

```
22088  121160  052777  000100  177054          BIS     #BIT6,@TCSR     ;ENABLE INTERRUPTS
22089  121166  000240                          NOP
22090
22091  121170  104000                          EMT                     ;XMIT DID NOT INTERRUPT
22092
22093  121172  042777  000100  177042  4$:     BIC     #BIT6,@TCSR     ;DISABLE INTERRUPTS
22094  121200  022626                          CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22095  121202  010377  177044                  MOV     R3,@TVECT       ;RESTORE XMIT VECTOR
22096
22097
22098                                  ;*************************************************************************************
22099                                  ;TEST 611       TEST THAT XMIT INTERRUPTS DO NOT OCCUR WHEN DISABLED
22100                                  ;*************************************************************************************
22101  121206                          TS611:
22102  121206  042777  000100  177026          BIC     #BIT6,@TCSR     ;DISABLE INTERRUPTS
22103  121214  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22104  121220  017703  177026                  MOV     @TVECT,R3       ;SAVE XMIT VECTOR
22105  121224  012777  121252  177020          MOV     #2$,@TVECT      ;POINT XMIT VECTOR TO ERROR REPORT
22106  121232  105777  177004          1$:     TSTB    @TCSR           ;WAIT FOR DONE
22107  121236  100375                          BPL     1$
22108  121240  052777  000100  176774          BIS     #BIT6,@TCSR     ;ENABLE INTERRUPT
22109  121246  000240                          NOP
22110  121250  000401                          BR      3$
22111  121252                          2$:
22112  121252  104000                          EMT                     ;
22113  121254  042777  000100  176760  3$:     BIC     #BIT6,@TCSR     ;CLEAR INTERRUPT ENABLE
22114  121262  012777  121300  176762          MOV     #4$,@TVECT      ;POINT XMIT VECTOR TO ERROR REPORT
22115  121270  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22116  121274  000240                          NOP
22117  121276  000401                          BR      5$
22118  121300                          4$:
22119  121300  104000                          EMT                     ;
22120  121302  010377  176744          5$:     MOV     R3,@TVECT       ;RESTORE XMIT VECTOR
22121
22122
22123                                  ;*************************************************************************************
22124                                  ;TEST 612       TEST TRANSMITTER FOR DOUBLE INTERRUPTS
22125                                  ;*************************************************************************************
22126  121306                          TS612:
22127  121306  042777  000100  176726          BIC     #BIT6,@TCSR     ;CLEAR INTERRUPT ENABLE
22128  121314  017703  176732                  MOV     @TVECT,R3       ;SAVE XMIT VECTOR
22129  121320  017704  176730                  MOV     @TPSW,R4        ;SAVE XMIT PSW VECTOR
22130  121324  012777  121364  176720          MOV     #2$,@TVECT      ;SET UP XMIT VECTOR
22131  121332  012777  000340  176714          MOV     #340,@TPSW      ;SET PIO 7 AFTER INTERRUPT
22132  121340  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22133  121344  105777  176672          1$:     TSTB    @TCSR           ;WAIT FOR DONE
22134  121350  100375                          BPL     1$
22135  121352  052777  000100  176662          BIS     #BIT6,@TCSR     ;ENABLE INTERRUPTS
22136  121360  000240                          NOP
22137
22138  121362  104000                          EMT                     ;
22139                                                                  ;XMIT INTERRUPT DID NOT OCCUR
22140  121364  022626                  2$:     CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22141  121366  012777  121412  176656          MOV     #4$,@TVECT      ;POINT XMIT VECTOR TO ERROR
22142  121374  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22143  121400  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPTS
```

```
22144  121402  042777  000100  176632          BIC     #BIT6,@TCSR     ;DISABLE INTERRUPTS
22145  121410  000401                          BR      5$
22146  121412                          4$:
22147  121412  104000                          EMT                     ;
22148  121414  010377  176632          5$:     MOV     R3,@TVECT       ;RESTORE XMIT VECTOR
22149  121420  010477  176630                  MOV     R4,@TPSW        ;RESTORE XMIT PSW VECTOR
22150
22151                                  ;*****************************************************************************
22152                                  ;TEST 613       TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF
22153                                  ;*****************************************************************************
22154  121424                          TS613:
22155  121424  042777  000100  176610          BIC     #BIT6,@TCSR     ;DISABLE INTERRUPTS
22156  121432  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22157  121436  017703  176610                  MOV     @TVECT,R3       ;SAVE XMIT VECTOR
22158  121442  012777  121512  176602          MOV     #2$,@TVECT      ;POINT XMIT VECTOR TO ERROR
22159  121450  052777  000100  176564          BIS     #BIT6,@TCSR     ;ENABLE INTERRUPTS
22160  121456  005077  176562                  CLR     @TBUF           ;LOAD TBUF
22161  121462  105777  176554          1$:     TSTB    @TCSR           ;WAIT FOR DONE (INTERRUPT)
22162  121466  100375                          BPL     1$
22163  121470  005077  176550                  CLR     @TBUF           ;FILL SECOND BUFFER TO RESET INT.
22164  121474  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22165  121500  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPTS
22166  121502  042777  000100  176532          BIC     #BIT6,@TCSR     ;DISABLE INTERRUPTS
22167  121510  000401                          BR      3$
22168  121512                          2$:
22169  121512  104000                          EMT                     ;
22170  121514  010377  176532          3$:     MOV     R3,@TVECT       ;RESTORE XMIT VECTOR
22171  121520  005000                          CLR     R0              ;INITIALIZE LOOP COUNTER
22172  121522  005200                  4$:     INC     R0              ;INCREMENT LOOP COUNTER
22173  121524  001376                          BNE     4$              ; UNTIL COUNTER = 0
22174  121526  005777  176506                  TST     @RBUF           ;CLEAR RECEIVER BUFFER
22175
22176
22177                                  ;*****************************************************************************
22178                                  ;TEST 614       TEST THAT RCVR DONE (7) SET & CLEAR PROPERLY
22179                                  ;*****************************************************************************
22180  121532                          TS614:
22181  121532  052777  000004  176502          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22182  121540  005000                          CLR     R0              ;CLEAR A TIMER
22183  121542  005077  176476                  CLR     @TBUF           ;LOAD TRANSMIT BUFFER
22184  121546  105777  176464          WDONE1: TSTB    @RCSR           ;CHECK FOR RECEIVER DONE
22185  121552  100406                          BMI     6$              ;BR, IF DONE
22186  121554  005200                          INC     R0              ;INCREMENT TIMER, IF NOT DONE
22187  121556  001373                          BNE     WDONE1          ;CONTINUE WAIT IF TIME REMAINS
22188  121560  042777  000004  176454          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22189  121566  104000                          EMT                     ;RECEIVER DONE NEVER SET
22190
22191  121570  000005                  6$:     RESET                   ;CLEAR DONE, MAINTENANCE MODE AND RECEIVER
22192                                                                  ;   BUFFER WITH RESET
22193  121572  105777  176440                  TSTB    @RCSR           ;CHECK FOR DONE CLEAR
22194  121576  001401                          BEQ     7$
22195
22196  121600  104000                          EMT                     ;RESET DID NOT CLEAR RCVR DONE
22197
22198  121602                          7$:
22199
```

```
22200                                   ;*****************************************************************************
22201                                   ;TEST 615       TEST THAT READING RBUF CLEARS RECEIVER DONE
22202                                   ;*****************************************************************************
22203  121602                           TS615:
22204  121602  052777  000004  176432          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22205  121610  005077  176430                  CLR     @TBUF           ;LOAD TRANSMITTER
22206  121614  105777  176416          1$:     TSTB    @RCSR           ;WAIT FOR RECEIVER DONE
22207  121620  100375                          BPL     1$
22208  121622  017700  176412                  MOV     @RBUF,R0        ;READ RECEIVE BUFFER
22209  121626  042777  000004  176406          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22210  121634  105777  176376                  TSTB    @RCSR           ;CHECK FOR RECEIVE DONE CLEAR
22211  121640  001401                          BEQ     TS616
22212  121642  104000                          EMT                     ;
22213                                                                  ;READING RBUF DID NOT CLEAR RCVR DONE
22214
22215
22216                                   ;*****************************************************************************
22217                                   ;TEST 616       TEST THAT RCVR INTERRUPTS ONLY WHEN ENABLED
22218                                   ;*****************************************************************************
22219  121644                           TS616:
22220  121644  042777  000100  176370          BIC     #BIT6,@TCSR     ;DISABLE TRANSMIT INTERRUPTS
22221  121652  042777  000100  176356          BIC     #BIT6,@RCSR     ;DISABLE RECEIVER INTERRUPTS
22222  121660  052777  000004  176354          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22223  121666  017703  176354                  MOV     @RVECT,R3       ;SAVE RECEIVE VECTOR
22224  121672  012777  121726  176346          MOV     #2$,@RVECT      ;POINT RCV VECTOR TO ERROR REPORT
22225  121700  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22226  121704  005077  176334                  CLR     @TBUF           ;SEND A CHARACTER
22227  121710  105777  176322          1$:     TSTB    @RCSR           ;WAIT FOR RECEIVER DONE
22228  121714  100375                          BPL     1$
22229  121716  042777  000004  176316          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22230  121724  000404                          BR      3$              ;CONTINUE TEST
22231
22232  121726  042777  000004  176306  2$:     BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22233  121734  104000                          EMT                     ;
22234  121736  012777  121756  176302  3$:     MOV     #4$,@RVECT      ;POINT RCV VECTOR TO END OF TEST
22235  121744  052777  000100  176264          BIS     #BIT6,@RCSR     ;ENABLE RCV INTERRUPTS
22236  121752  000240                          NOP                     ;GIVE ANY INTERRUPTS TIME
22237  121754  104000                          EMT                     ;
22238  121756  042777  000100  176252  4$:     BIC     #BIT6,@RCSR     ;DISABLE INTERRUPTS
22239  121764  022626                          CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22240  121766  010377  176254                  MOV     R3,@RVECT       ;RESTORE RECEIVE VECTOR
22241  121772  005777  176242                  TST     @RBUF           ;CLEAR RECEIVER BUFFER
22242
22243
22244                                   ;*****************************************************************************
22245                                   ;TEST 617       TEST THAT RCVR INTERRUPTS DO NOT OCCUR WHEN DISABLED
22246                                   ;*****************************************************************************
22247  121776                           TS617:
22248  121776  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22249  122002  017703  176240                  MOV     @RVECT,R3       ;SAVE RECEIVE VECTOR
22250  122006  012777  122046  176232          MOV     #2$,@RVECT      ;POINT RCVR VECTOR TO ERROR REPORT
22251  122014  052777  000004  176220          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22252  122022  005077  176216                  CLR     @TBUF           ;SEND A CHARACTER
22253  122026  105777  176204          1$:     TSTB    @RCSR           ;WAIT FOR RECEIVER DONE
22254  122032  100375                          BPL     1$
22255  122034  052777  000100  176174          BIS     #BIT6,@RCSR     ;ENABLE INTERRUPTS
```

```
22256  122042  000240                          NOP                     ;GIVE TIME FOR INTERRUPT
22257  122044  000404                          BR      3$              ;CONTINUE TEST
22258  122046  042777  000004  176166  2$:     BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22259  122054  104000                          EMT                     ;
22260  122056  042777  000100  176152  3$:     BIC     #BIT6,@RCSR     ;CLEAR INTERRUPT ENABLE
22261  122064  012777  122110  176154          MOV     #4$,@RVECT      ;POINT RCVR VECTOR TO ERROR REPORT
22262  122072  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22263  122076  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPT
22264  122100  042777  000004  176134          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22265  122106  000404                          BR      5$              ;BR TO END OF TEST, IF NO INTERRUPT
22266
22267  122110  042777  000004  176124  4$:     BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22268  122116  104000                          EMT                     ;
22269  122120  010377  176122          5$:     MOV     R3,@RVECT       ;RESTORE RECEIVE VECTOR
22270  122124  005777  176110                  TST     @RBUF           ;CLEAR RECEIVER BUFFER
22271
22272
22273                                  ;*********************************************************************************
22274                                  ;TEST 620       TEST RECEIVER FOR DOUBLE INTERRUPTS
22275                                  ;*********************************************************************************
22276  122130                          TS620:
22277  122130  017703  176112                  MOV     @RVECT,R3       ;SAVE RECEIVE VECTOR
22278  122134  017704  176110                  MOV     @RPSW,R4        ;SAVE RECEIVE PSW VECTOR
22279  122140  012777  122220  176100          MOV     #2$,@RVECT      ;POINT RCV VECTOR TO CONTINUE TEST
22280  122146  012777  000340  176074          MOV     #340,@RPSW      ;SET PRIORITY TO 7 AFTER INTERRUPT
22281  122154  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22282  122160  052777  000004  176054          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22283  122166  005077  176052                  CLR     @TBUF           ;SEND A CHARACTER
22284  122172  105777  176040          1$:     TSTB    @RCSR           ;WAIT FOR RCVR DONE
22285  122176  100375                          BPL     1$
22286  122200  042777  000004  176034          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22287  122206  052777  000100  176022          BIS     #BIT6,@RCSR     ;ENABLE RCV INTERRUPTS
22288  122214  000240                          NOP                     ;GIVE SOME TIME
22289  122216  104000                          EMT                     ;
22290  122220  022626          2$:     CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22291  122222  012777  122256  176016          MOV     #3$,@RVECT      ;POINT RCV VECTOR TO ERROR REPORT
22292  122230  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 7
22293  122234  000240                          NOP                     ;GIVE SOME TIME
22294  122236  042777  000100  175772          BIC     #BIT6,@RCSR     ;CLEAR INTERRUPT ENABLE
22295  122244  010377  175776                  MOV     R3,@RVECT       ;RESTORE RECEIVE VECTOR
22296  122250  010477  175774                  MOV     R4,@RPSW        ;RESTORE RECEIVE PSW VECTOR
22297  122254  000401                          BR      4$
22298  122256                          3$:
22299  122256  104000                          EMT                     ;
22300  122260  010377  175762          4$:     MOV     R3,@RVECT       ;RESTORE RECEIVE VECTOR
22301  122264  005777  175750                  TST     @RBUF           ;CLEAR RECEIVER BUFFER
22302
22303
22304                                  ;*********************************************************************************
22305                                  ;TEST 621       TEST THAT RCVR INTERRUPT CLEARS BY READING RBUF
22306                                  ;*********************************************************************************
22307  122270                          TS621:
22308  122270  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22309  122274  017703  175746                  MOV     @RVECT,R3       ;SAVE RECEIVE VECTOR
22310  122300  012777  122364  175740          MOV     #2$,@RVECT      ;POINT RCV VECTOR TO ERROR REPORT
22311  122306  052777  000100  175722          BIS     #BIT6,@RCSR     ;SET RCVR INTERRUPT ENABLE
```

```
22312  122314  052777  000004  175720          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22313  122322  005077  175716                  CLR     @TBUF           ;SEND A CHARACTER
22314  122326  105777  175704          1$:     TSTB    @RCSR           ;WAIT FOR DONE (INTERRUPT)
22315  122332  100375                          BPL     1$
22316  122334  042777  000004  175700          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22317  122342  005777  175672                  TST     @RBUF           ;READ RBUF TO CLEAR PENDING INTERRPUT
22318  122346  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22319  122352  000240                          NOP                     ;ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
22320  122354  042777  000100  175654          BIC     #BIT6,@RCSR     ;NO INTERRUPT-CLEAR INT. ENABLE
22321  122362  000401                          BR      3$
22322  122364                          2$:
22323  122364  104000                          EMT                     ;
22324  122366  010377  175654          3$:     MOV     R3,@RVECT       ;RESTORE RECEIVE VECTOR
22325
22326
22327
22328                                   ;*****************************************************************************
22329                                   ;TEST 622       TEST THAT RESET CLEARS RECEIVE INTERRUPT
22330                                   ;*****************************************************************************
22331  122372                           TS622:
22332  122372  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22333  122376  017703  175644                  MOV     @RVECT,R3       ;SAVE RECEIVE VECTOR
22334  122402  012777  122466  175636          MOV     #2$,@RVECT      ;POINT RCV VECTOR TO ERROR REPORT
22335  122410  052777  000100  175620          BIS     #BIT6,@RCSR     ;SET RCV INTERRUPT ENABLE
22336  122416  052777  000004  175616          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22337  122424  012777  000377  175612          MOV     #377,@TBUF      ;SEND AN ALL 1'S CHARACTER
22338  122432  105777  175600          1$:     TSTB    @RCSR           ;WAIT FOR RCV DONE
22339  122436  100375                          BPL     1$
22340  122440  000005                          RESET                   ;CLEAR RCV INTERRUPT & RBUF
22341  122442  052777  000100  175566          BIS     #BIT6,@RCSR     ;TURN ON RECEIVER INTERRUPT
22342  122450  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22343  122454  000240                          NOP                     ;ALLOW TIME FOR AN ERRONEOUS INTERRUPT
22344  122456  042777  000100  175552          BIC     #BIT6,@RCSR     ;NO INTERRUPT-CLEAR INT. ENABLE
22345  122464  000401                          BR      3$
22346  122466                          2$:
22347  122466  104000                          EMT                     ;
22348  122470  010377  175552          3$:     MOV     R3,@RVECT       ;RESTORE RECEIVE VECTOR
22349
22350
22351                                   ;*****************************************************************************
22352                                   ;TEST 623       TEST THAT THE "OR" ERROR (BIT14) & "ERROR" (BIT15) CAN BE SET
22353                                   ;*****************************************************************************
22354  122474                           TS623:
22355  122474  052777  000004  175540          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22356  122502  012700  000003                  MOV     #3,R0           ;SET CHARACTER COUNT TO SEND 3 CHAR.
22357  122506  005077  175532          1$:     CLR     @TBUF           ;LOAD TRANSMIT BUFFER
22358  122512  105777  175524          2$:     TSTB    @TCSR           ;WAIT FOR TRANSMIT DONE
22359  122516  100375                          BPL     2$
22360  122520  005300                          DEC     R0              ;DECREMENT CHARACTER COUNT
22361  122522  001371                          BNE     1$              ;BR IF ALL CHARACTERS NOT TRANSMITTED
```

```
22362  122524  042777  000004  175510          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22363  122532  032777  040000  175500          BIT     #BIT14,@RBUF    ;TEST FOR "OR" ERROR FLAG
22364  122540  001001                          BNE     3$
22365  122542  104000                          EMT                     ;
22366  122544  032777  100000  175466  3$:     BIT     #BIT15,@RBUF    ;TEST "ERROR" FLAG
22367  122552  001001                          BNE     4$
22368  122554  104000                          EMT                     ;
22369  122556  005000                  4$:     CLR     R0              ;INITIALIZE LOOP COUNTER
22370  122560  005200                  5$:     INC     R0              ;INCREMENT LOOP COUNTER
22371  122562  001376                          BNE     5$              ;  UNTIL COUNTER = 0
22372  122564  005777  175450                  TST     @RBUF           ;CLEAR RECEIVER BUFFER
22373
22374
22375                                  ;*********************************************************************************
22376                                  ;TEST 624       TEST DATA PATH FROM TRANSMITTER TO RECEIVER USING MAINTENANCE WRAP
22377                                  ;*********************************************************************************
22378  122570                          TS624:
22379  122570  000005                          RESET                   ;CLEAR THE WORLD
22380  122572  005001                          CLR     R1              ;CLEAR REGISTER FOR TEST DATA
22381  122574  052777  000004  175440          BIS     #BIT2,@TCSR     ;SET MAINTENANCE WRAP
22382  122602  105201                  1$:     INCB    R1              ;INCREMENT THE TEST DATA
22383  122604  110177  175434                  MOVB    R1,@TBUF        ;XMIT A CHARACTER
22384  122610  105777  175422          2$:     TSTB    @RCSR           ;WAIT FOR RECEIVER DONE
22385  122614  100375                          BPL     2$
22386  122616  017702  175416                  MOV     @RBUF,R2        ;GET RECEIVED CHARACTER
22387  122622  020102                          CMP     R1,R2           ;COMPARE DATA
22388  122624  001404                          BEQ     3$              ;BR, IF NON-COMPARE
22389  122626  042777  000004  175406          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22390  122634  104000                          EMT                     ;
22391  122636  105701                  3$:     TSTB    R1              ;TEST XMIT DATA FOR ZERO
22392  122640  001360                          BNE     1$              ;IF ALL DATA HES NOT BEEN CHECKED GO BACK
22393  122642  042777  000004  175372          BIC     #BIT2,@TCSR     ;CLEAR MAINTENANCE BIT
22394  122650  000167  000016                  JMP     KWSTRT          ;GET TO NEXT TEST
22395
22396  122654  012737  000004  001002  ERROR5: MOV     #4,@#$FATAL     ;SET UP FATAL ERROR NUMBER
22397  122662  012767  000001  056110          MOV     #1,$MSGTY       ;SET FATAL ERROR FLAG
22398  122670  000777                  SL1HLT: BR      .
22399
22400
22401  122672  032777  000010  076364  KWSTRT: BIT     #10,@SWR
22402  122700  001402                          BEQ     1$
22403  122702  000167  001214                  JMP     SLU2ST
22404  122706  012737  000005  001004  1$:     MOV     #5,@#$TESTN     ;PUT TEST NUMBER IN MAILBOX
22405  122714  012737  124064  000030          MOV     #ERROR6,@#30    ;SET UP ERROR CALL
22406
22407  122722                          LKSTST:
22408                                  ;*********************************************************************************
22409                                  ;TEST 625       TEST ABILITY TO ACCESS LKS
22410                                  ;*********************************************************************************
22411  122722                          TS625:
22412  122722  013703  000004                  MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
22413  122726  012737  122742  000004          MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
22414  122734  005777  175316                  TST     @LKS            ;ACCESS LKS
22415  122740  000401                          BR      2$
22416  122742                          1$:
22417  122742  104000                          EMT                     ;
```

```
22418  122744  010337  000004          2$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
22419
22420                                   ;*****************************************************************************
22421                                   ;TEST 626       TEST THAT BIT6 OF LKS CAN BE SET & RESET
22422                                   ;*****************************************************************************
22423  122750                           TS626:
22424  122750  017703  175304                  MOV     @RTCVT,R3       ;SAVE LINE CLOCK VECTOR
22425  122754  012777  122776  175276        MOV     #1$,@RTCVT      ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
22426  122762  106427  000340                MTPS    #340            ;SET PSW TO PRIORITY 7
22427  122766  032777  000100  175262        BIT     #BIT6,@LKS      ;TEST BIT6 OF LKS
22428  122774  001401                        BEQ     2$
22429  122776                           1$:
22430  122776  104000                        EMT                     ;
22431  123000  052777  000100  175250  2$:   BIS     #BIT6,@LKS      ;SET BIT6 OF LKS
22432  123006  032777  000100  175242        BIT     #BIT6,@LKS      ;TEST BIT6 OF LKS
22433  123014  001001                        BNE     3$
22434  123016  104000                        EMT                     ;
22435  123020  042777  000100  175230  3$:   BIC     #BIT6,@LKS      ;CLEAR BIT6 OF LKS
22436  123026  032777  000100  175222        BIT     #BIT6,@LKS      ;TEST BIT6 OF LK
22437  123034  001401                        BEQ     4$
22438  123036  104000                        EMT                     ;
22439  123040  032737  000001  001020  4$:   BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
22440  123046  001403                        BEQ     70$             ;IF NO THEN DO TEST
22441  123050  005737  001006                TST     @#$PASS         ;IS THIS FIRST PASS
22442  123054  001011                        BNE     5$              ;IF NO SKIP TO TEST END
22443  123056                           70$:
22444  123056  052777  000100  175172        BIS     #BIT6,@LKS      ;SET BIT6 OF LKS
22445  123064  000005                        RESET                   ;CLEAR BIT6 OF LKS WITH RESET
22446  123066  032777  000100  175162        BIT     #BIT6,@LKS      ;TEST BIT6 OF LKS
22447  123074  001401                        BEQ     5$
22448  123076  104000                        EMT                     ;
22449  123100  010377  175154          5$:   MOV     R3,@RTCVT       ;RESTORE LINE CLOCK VECTOR
22450
22451                                   ;*****************************************************************************
22452                                   ;TEST 627       TEST THAT BIT7 OF LKS SETS & CAN BE CLEARED
22453                                   ;*****************************************************************************
22454  123104                           TS627:
22455  123104                           1$:
22456  123104  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
22457  123112  001403                        BEQ     70$             ;IF NO THEN DO TEST
22458  123114  005737  001006                TST     @#$PASS         ;IS THIS FIRST PASS
22459  123120  001032                        BNE     TS630           ;IF NO THEN SHIP TO NEXT TEST
22460  123122                           70$:
22461  123122  105777  175130                TSTB    @LKS            ;TEST FOR BIT7 OF LKS
22462  123126  100401                        BMI     2$
22463  123130  104000                        EMT                     ;
22464  123132  042777  000200  175116  2$:   BIC     #BIT7,@LKS      ;CLEAR BIT7 OF LKS
22465  123140  032777  000200  175110        BIT     #BIT7,@LKS      ;TEST BIT7 OF LKS
22466  123146  001410                        BEQ     3$
22467  123150  042777  000200  175100        BIC     #BIT7,@LKS      ;TRY ONE MOR TIME BECAUSE THE CLOCK MAY
22468  123156  032777  000200  175072        BIT     #BIT7,@LKS      ;MAY HAVE SET IMMEDIATELY AFTER THE FIRST CLEAR
22469  123164  001401                        BEQ     3$
22470  123166  104000                        EMT                     ;
22471  123170  005000                  3$:   CLR     R0              ;CLEAR TIMER
22472  123172  105777  175060          CONT1: TSTB   @LKS            ;TEST FOR BIT7 OF LKS
22473  123176  100403                        BMI     1$              ;BR, IF SET
```

```
22474  123200  005200                          INC     R0              ;INCREMENT TIMER
22475  123202  001373                          BNE     CONT1
22476  123204  104000                          EMT                     ;
22477  123206                          1$:
22478
22479
22480                                  ;*************************************************************************************
22481                                  ;TEST 630       TEST THAT THE REAL TIME CLOCK INTERRUPTS PROPERLY
22482                                  ;*************************************************************************************
22483  123206                          TS630:
22484  123206  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22485  123212  017703  175042                  MOV     @RTCVT,R3       ;SAVE LINE CLOCK VECTOR
22486  123216  017704  175040                  MOV     @RTCPSW,R4      ;SAVE LINE CLOCK PSW VECTOR
22487  123222  012777  123264  175030          MOV     #2$,@RTCVT      ;SET RTC INTERRUPT VECTOR TO ERROR REPORT
22488  123230  012777  000340  175024          MOV     #340,@RTCPSW    ;KEEP PRIORITY AT 7
22489  123236  042777  000200  175012          BIC     #BIT7,@LKS      ;CLEAR CLOCK DONE FLAG
22490  123244  052777  000100  175004          BIS     #BIT6,@LKS      ;SET INTERRUPT ENABLE
22491  123252  105777  175000          1$:     TSTB    @LKS            ;WAIT FOR RTC DONE(INTERRUPT REQUEST)
22492  123256  100375                          BPL     1$
22493  123260  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPTS
22494  123262  000401                          BR      3$
22495  123264                          2$:
22496  123264  104000                          EMT                     ;
22497  123266  005077  174764          3$:     CLR     @LKS            ;DISABLE RTC INTERRUPTS & CLEAR DONE
22498  123272  012777  123316  174760          MOV     #4$,@RTCVT      ;SET RTC INTERRUPT VECTOR FOR ERROR
22499  123300  106427  000240                  MTPS    #240            ;CHANGE PSW TO PRIORITY 5
22500  123304  105777  174746          20$:    TSTB    @LKS            ;WAIT FOR DONE (INTERRUPT REQUEST)
22501  123310  100375                          BPL     20$
22502  123312  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPT
22503  123314  000401                          BR      5$
22504  123316                          4$:
22505  123316  104000                          EMT                     ;
22506  123320  012777  123354  174732  5$:     MOV     #7$,@RTCVT      ;POINT RTC VECTOR TO END OF TEST
22507  123326  042777  000200  174722          BIC     #BIT7,@LKS      ;CLEAR CLOCK DONE FLAG
22508  123334  052777  000100  174714          BIS     #BIT6,@LKS      ;ALLOW INTERRUPTS
22509  123342  105777  174710          6$:     TSTB    @LKS            ;WAIT FOR RTC DONE
22510  123346  100375                          BPL     6$
22511  123350  000240                          NOP                     ;GIVE TIME FOR INTERRUPT
22512
22513  123352  104000                          EMT                     ;RTC INTERRUPT DID NOT OCCUR
22514
22515  123354  022626                  7$:     CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22516  123356  042777  000100  174672          BIC     #BIT6,@LKS      ;DISABLE INTERRUPTS
22517  123364  010377  174670                  MOV     R3,@RTCVT       ;RESTORE LINE CLOCK VECTOR
22518  123370  010477  174666                  MOV     R4,@RTCPSW      ;RESTORE LINE CLOCK PSW VECTOR
22519
22520
22521
22522
22523
22524                                  ;*************************************************************************************
22525                                  ;TEST 631       TEST RTC FOR DOUBLE INTERRUPTS
22526                                  ;*************************************************************************************
22527  123374                          TS631:
22528  123374  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
22529  123402  001403                          BEQ     70$             ;IF NO THEN DO TEST
```

```
22530  123404  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
22531  123410  001047                          BNE     TS632           ;IF NO THEN SHIP TO NEXT TEST
22532  123412                                  70$:
22533  123412  017703  174642                  MOV     @RTCVT,R3       ;SAVE LINE CLOCK VECTOR
22534  123416  017704  174640                  MOV     @RTCPSW,R4      ;SAVE LINE CLOCK PSW VECTOR
22535  123422  012777  123470  174630          MOV     #2$,@RTCVT      ;SET UP RTC INTERRUPT VECTOR
22536  123430  012777  000340  174624          MOV     #340,@RTCPSW    ;DISALLOW INTERRUPTS AFTER THE INTERRUPT
22537  123436  106427  000240                  MTPS    #240            ;SET PSW TO PRIORITY 5
22538  123442  042777  000200  174606          BIC     #BIT7,@LKS      ;CLEAR CLOCK DONE FLAG
22539  123450  052777  000100  174600          BIS     #BIT6,@LKS      ;ENABLE CLOCK INTERRUPTS
22540  123456  105777  174574                  1$:     TSTB    @LKS            ;WAIT FOR DONE
22541  123462  100375                          BPL     1$
22542  123464  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPT
22543
22544  123466  104000                          EMT                     ;RTC INTERRUPT DID NOT OCCUR
22545
22546  123470  022626                  2$:     CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22547  123472  012777  123510  174560          MOV     #3$,@RTCVT      ;POINT RTC VECTOR TO ERROR REPORT
22548  123500  106427  000240                  MTPS    #240            ;SET PSW TO PRIORITY 5
22549  123504  000240                          NOP                     ;GIVE SOME TIME FOR AN INTERRUPT
22550  123506  000401                          BR      4$
22551  123510                                  3$:
22552  123510  104000                          EMT                     ;
22553  123512  042777  000100  174536  4$:     BIC     #BIT6,@LKS      ;DISABLE CLOCK INTERRUPTS
22554  123520  010377  174534                  MOV     R3,@RTCVT       ;RESTORE LINE CLOCK VECTOR
22555  123524  010477  174532                  MOV     R4,@RTCPSW      ;RESTORE LINE CLOCK PSW VECTOR
22556
22557
22558                                  ;*******************************************************************************************
22559                                  ;TEST 632       TEST THAT RTC INTERRUPT CLEARS WITH RESET
22560                                  ;*******************************************************************************************
22561  123530                          TS632:
22562  123530  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
22563  123536  001403                          BEQ     70$             ;IF NO THEN DO TEST
22564  123540  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
22565  123544  001033                          BNE     TS633           ;IF NO THEN SHIP TO NEXT TEST
22566  123546                                  70$:
22567  123546  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22568  123552  017703  174502                  MOV     @RTCVT,R3       ;SAVE LINE CLOCK VECTOR
22569  123556  012777  123626  174474          MOV     #2$,@RTCVT      ;POINT RTC VECTOR TO ERROR REPORT
22570  123564  042777  000200  174464          BIC     #BIT7,@LKS      ;CLEAR CLOCK DONE FLAG
22571  123572  052777  000100  174456          BIS     #BIT6,@LKS      ;ENABLE CLOCK INTERRUPTS
22572  123600  105777  174452                  1$:     TSTB    @LKS            ;WAIT FOR DONE (INTERRUPT REQUEST)
22573  123604  100375                          BPL     1$
22574  123606  000005                          RESET                   ;CLEAR PENDING INTERRUPT WITH RESET
22575  123610  106427  000240                  MTPS    #240            ;SET PSW TO PRIORITY 5
22576  123614  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPT
22577  123616  042777  000100  174432          BIC     #BIT6,@LKS      ;DISALLOW INTERRUPTS
22578  123624  000401                          BR      3$
22579  123626                                  2$:
22580  123626  104000                          EMT                     ;
22581  123630  010377  174424                  3$:     MOV     R3,@RTCVT       ;RESTORE LINE CLOCK VECTOR
22582
22583
22584                                  ;*******************************************************************************************
22585                                  ;TEST 633       TEST THAT RTC INTERRUPT CLEARS BY CLEARING BIT7 OF LKS
```

```
22586
22587  123634
22588  123634  106427  000340                  ;*****************************************************************************
22589  123640  017703  174414          TS633:
22590  123644  012777  123720  174406          MTPS    #340            ;SET PSW TO PRIORITY 7
22591  123652  042777  000200  174376          MOV     @RTCVT,R3       ;SAVE LINE CLOCK VECTOR
22592  123660  052777  000100  174370          MOV     #2$,@RTCVT      ;POINT RTC VECTOR TO ERROR REPORT
22593  123666  105777  174364          1$:     BIC     #BIT7,@LKS      ;CLEAR CLOCK DONE FLAG
22594  123672  100375                          BIS     #BIT6,@LKS      ;ENABLE CLOCK INTERRUPTS
22595  123674  042777  000200  174354          TSTB    @LKS            ;WAIT FOR DONE (INTERRUPT REQUEST)
22596  123702  106427  000240                  BPL     1$
22597  123706  000240                          BIC     #BIT7,@LKS      ;CLEAR DONE & INTERRUPT
22598  123710  042777  000100  174340          MTPS    #240            ;SET PSW TO PRIORITY 5
22599  123716  000401                          NOP                     ;GIVE TIME FOR ANY INTERRUPT
22600  123720                                  BIC     #BIT6,@LKS      ;DISALLOW INTERRUPTS
22601  123720  104000                          BR      3$
22602                                  2$:
22603  123722  010377  174332                  EMT                     ;
22604  123726  106427  000340
22605                                  3$:     MOV     R3,@RTCVT       ;RESTORE LINE CLOCK VECTOR
22606                                          MTPS    #340            ;SET PSW TO PRIORITY 7
22607
22608                                  ;*****************************************************************************
22609                                  ;TEST 634       TEST CLOCK REPEATABILITY
22610  123732                          ;*****************************************************************************
22611  123732  032737  000001  001020  TS634:
22612  123740  001403                          BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
22613  123742  005737  001006                  BEQ     70$             ;IF NO GO DO TEST
22614  123746  001065                          TST     @#$PASS         ;IS THIS FIRST PASS
22615  123750                                  BNE     SLU2ST          ;IF NO THEN SKIP REST OF RTC TESTING
22616  123750  042777  000100  174300  70$:
22617                                          BIC     #BIT6,@LKS      ;DISALLOW INTERRUPTS
22618  123756  005000
22619  123760  012701  177777                  CLR     R0              ;CLEAR A TIMER
22620  123764  005002                  1$:     MOV     #-1,R1          ;SET A FLAG INDICATING FIRST PASS THRU THIS LOOP
22621  123766  005077  174264                  CLR     R2              ;CLEAR CLOCK COUNTER
22622  123772  105777  174260          2$:     CLR     @LKS            ;CLEAR DONE
22623  123776  100375                          TSTB    @LKS            ;SYNC ON DONE
22624  124000  005077  174252                  BPL     2$
22625  124004  105777  174246          3$:     CLR     @LKS            ;CLEAR DONE
22626  124010  100003                          TSTB    @LKS            ;IS CLOCK DONE?
22627  124012  005202                          BPL     4$              ;BR IF NOT , TO INCREMENT TIMER
22628  124014  005077  174236                  INC     R2              ;IF DONE, INCREMENT CLOCK COUNT
22629  124020  005200                  4$:     CLR     @LKS            ;CLEAR DONE
22630  124022  001370                          INC     R0              ;INCREMENT TIMER
22631  124024  005201                          BNE     3$              ;BR IF TIME REMAINS
22632  124026  001003                          INC     R1              ;INCREMENT LOOP PASS FLAG
22633  124030  010267  000024                  BNE     CMPARE          ;BR IF TWO PASSES HAVE BEEN MADE
22634  124034  000753                          MOV     R2,KFIRST       ;IF NOT, STORE FIRST CLOCK COUNT
22635  124036  016701  000016                  BR      1$              ;DO LOOP AGAIN
22636  124042  160201                  CMPARE: MOV     KFIRST,R1       ;RECALL FIRST CLOCK COUNT
22637  124044  100001                          SUB     R2,R1           ;CALCULATE DIFFERENCE OF TWO COUNTS
22638  124046  005401                          BPL     TOLER           ;IF POSITIVE,SKIP NEGATION OF DIFFERENCE
22639  124050  020127  000002          TOLER:  NEG     R1              ;MAKE DIFFERENCE A POSITIVE NUMBER
22640  124054  003422                          CMP     R1,#2           ;COMPARE DIFFERENCE WITH DESIRED TOLERANCE
22641  124056  104000                          BLE     SLU2ST
                                               EMT                     ;
```

```
22642
22643  124060  000000                  KFIRST: .WORD   0
22644  124062  000000                  SECND:  .WORD   0
22645
22646
22647  124064  012737  000005  001002  ERROR6: MOV     #5,@#$FATAL                     ;SET UP FATAL ERROR NUMBER
22648  124072  012767  000001  054700          MOV     #1,$MSGTY                       ;SET FATAL ERROR FLAG
22649  124100  000777                  LTCHLT: BR      .
22650
22651                                  ;SERIAL LINE UNIT REGISTER AND VECTOR ADDRESSES FOR SLU2
22652
22653  124102  176500                  RCSR2:  176500                  ;ADDRESS OF RECEIVER COMMAND/STATUS REGISTER
22654  124104  176502                  RBUF2:  176502                  ;ADDRESS OF RECEIVER BUFFER
22655  124106  176504                  TCSR2:  176504                  ;ADDRESS OF TRANSMITTER COMMAND/STATUS REGISTER
22656  124110  176506                  TBUF2:  176506                  ;ADDRESS OF TRANSMITTER BUFFER
22657  124112  000300                  RVECT2: 300                     ;RECEIVER INTERRUPT VECTOR
22658  124114  000302                  RPSW2:  302
22659  124116  000304                  TVECT2: 304                     ;TRANSMITTER INTERRUPT VECTOR
22660  124120  000306                  TPSW2:  306
22661
22662  124122  032777  000020  075134  SLU2ST: BIT     #20,@SWR
22663  124130  001402                          BEQ     1$
22664  124132  000167  002556                  JMP     UNIQUE
22665  124136  012737  000006  001004  1$:     MOV     #6,@#$TESTN             ;PUT TEST NUMBER IN MAILBOX
22666  124144  012737  126654  000030          MOV     #ERROR7,@#30            ;SET UP FOR CORRECT ERROR CALL
22667
22668
22669                                  ;*****************************************************************************
22670                                  ;TEST 635       TEST ABILITY TO REFERENCE TCSR2
22671                                  ;*****************************************************************************
22672  124152                          TS635:
22673  124152  013703  000004                  MOV     @#4,R3                  ;SAVE TIMEOUT VECTOR
22674  124156  012737  124172  000004          MOV     #1$,@#4                 ;SET UP TIMEOUT VECTOR
22675  124164  005777  177716                  TST     @TCSR2                  ;REFERENCE THE XMIT COMMAND/STATUS REG.
22676  124170  000401                          BR      4$
22677  124172                          1$:
22678  124172  104000                          EMT                             ;
22679  124174  010337  000004          4$:     MOV     R3,@#4                  ;RESTORE TIMEOUT VECTOR
22680
22681
22682
22683                                  ;*****************************************************************************
22684                                  ;TEST 636       TEST ABILITY TO REFERENCE TBUF2
22685                                  ;*****************************************************************************
22686  124200                          TS636:
22687  124200  013703  000004                  MOV     @#4,R3                  ;SAVE TIMEOUT VECTOR
22688  124204  012737  124220  000004          MOV     #1$,@#4                 ;SET UP TIMEOUT VECTOR
22689  124212  005777  177672                  TST     @TBUF2                  ;REFERENCE THE XMIT BUFFER
22690  124216  000401                          BR      4$
22691  124220                          1$:
22692  124220  104000                          EMT                             ;
22693  124222  010337  000004          4$:     MOV     R3,@#4                  ;RESTORE TIMEOUT VECTOR
22694
22695
22696                                  ;*****************************************************************************
22697                                  ;TEST 637       TEST THAT TCSR2 BIT7(DONE) CLEARS WHEN XBUF IS LOADED
```

```
22698
22699  124226
22700  124226  032737  000001  001020          ;*****************************************************************************
22701  124234  001403                          TS637:
22702  124236  005737  001006                          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
22703  124242  001022                          BEQ     70$             ;IF NO THEN DO TEST
22704  124244                                  TST     @#$PASS         ;IS THIS FIRST PASS
22705  124244  005077  177640                BNE     TS640           ;IF NO THEN SHIP TO NEXT TEST
22706  124250  105777  177632                70$:
22707  124254  100006                          CLR     @TBUF2          ;LOAD XBUF
22708                                          TSTB    @TCSR2          ;CHECK DONE
22709                                          BPL     3$              ;BR IF CLEAR
22710  124256  005077  177626                                          ;FILL SECOND BUFFER BECUASE REFRESH COULD CAUSE
22711  124262  105777  177620                                          ;FIRST TEST TO FAIL
22712  124266  100001                          CLR     @TBUF2          ;FILL DOUBLE BUFFER
22713  124270  104000                          TSTB    @TCSR2          ;CHECK DONE
22714  124272  005000                          BPL     3$
22715  124274  105777  177606                EMT                     ;
22716  124300  100403                  3$:     CLR     R0              ;CLEAR TIMER
22717  124302  005200                  4$:     TSTB    @TCSR2          ;CHECK FOR XMIT DONE
22718  124304  001373                          BMI     5$              ;IF DONE SETS, BR TO END OF TEST
22719  124306  104000                          INC     R0              ;INCREMENT TIMER
22720  124310                                  BNE     4$
22721                                          EMT                     ;
22722                                  5$:
22723                                  ;*****************************************************************************
22724                                  ;TEST 640       TEST THAT TCSR2 ''DONE'' SETS WITH RESET
22725                                  ;*****************************************************************************
22726  124310                          TS640:
22727  124310  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
22728  124316  001403                          BEQ     70$             ;IF NO THEN DO TEST
22729  124320  005737  001006                TST     @#$PASS         ;IS THIS FIRST PASS
22730  124324  001015                          BNE     TS641           ;IF NO THEN SHIP TO NEXT TEST
22731  124326                                  70$:
22732  124326  005077  177556                CLR     @TBUF2          ;LOAD TRANSMIT BUFFER
22733  124332  105777  177550                1$:     TSTB    @TCSR2          ;WAIT FOR DONE
22734  124336  100375                          BPL     1$
22735  124340  005077  177544                CLR     @TBUF2          ;LOAD SECOND BUFFER
22736  124344  000240                          NOP
22737  124346  000005                          RESET                   ;SET DONE WITH RESET
22738  124350  105777  177532                TSTB    @TCSR2          ;CHECK FOR DONE SET
22739  124354  100401                          BMI     TS641
22740  124356  104000                          EMT                     ;
22741
22742
22743
22744                                  ;*****************************************************************************
22745                                  ;TEST 641       TEST ABILITY TO ACCESS RCSR2
22746                                  ;*****************************************************************************
22747  124360                          TS641:
22748  124360  013703  000004                  MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
22749  124364  012737  124400  000004          MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
22750  124372  005777  177504                TST     @RCSR2          ;ACCESS RCSR
22751  124376  000401                          BR      2$
22752  124400                                  1$:
22753  124400  104000                          EMT                     ;
```

```
22754  124402  010337  000004          2$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
22755
22756
22757                                  ;*********************************************************************************
22758                                  ;TEST 642       TEST ABILITY TO ACCESS RBUF2
22759                                  ;*********************************************************************************
22760  124406                          TS642:
22761  124406  013703  000004                  MOV     @#4,R3          ;SAVE TIMEOUT VECTOR
22762  124412  012737  124426  000004          MOV     #1$,@#4         ;SET UP TIMEOUT VECTOR
22763  124420  005777  177460                  TST     @RBUF2          ;ACCESS RBUF
22764  124424  000401                          BR      2$
22765  124426                          1$:
22766  124426  104000                          EMT                     ;
22767  124430  010337  000004          2$:     MOV     R3,@#4          ;RESTORE TIMEOUT VECTOR
22768
22769
22770
22771
22772
22773
22774                                  ;*********************************************************************************
22775                                  ;TEST 643       TEST THAT BIT0(BREAK BIT) CAN BE SET & CLEARED & RESET
22776                                  ;*********************************************************************************
22777  124434                          TS643:
22778  124434  032777  000001  177444          BIT     #BIT0,@TCSR2    ;CHECK BIT0 OF TCSR CLEAR
22779  124442  001401                          BEQ     3$
22780  124444  104000                          EMT                     ;
22781  124446  052777  000001  177432  3$:     BIS     #BIT0,@TCSR2    ;SET BIT0 IN TCSR
22782  124454  032777  000001  177424          BIT     #BIT0,@TCSR2    ;TEST BIT0 OF TCSR
22783  124462  001001                          BNE     4$
22784  124464  104000                          EMT                     ;
22785  124466  042777  000001  177412  4$:     BIC     #BIT0,@TCSR2    ;CLEAR BIT0 OF TCSR
22786  124474  032777  000001  177404          BIT     #BIT0,@TCSR2    ;TEST BIT0 OF TCSR
22787  124502  001401                          BEQ     7$
22788  124504  104000                          EMT                     ;
22789  124506                          7$:
22790  124506  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
22791  124514  001403                          BEQ     70$             ;IF NO THEN DO TEST
22792  124516  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
22793  124522  001011                          BNE     TS644           ;IF NO THEN SHIP TO NEXT TEST
22794  124524                          70$:
22795  124524  052777  000001  177354          BIS     #BIT0,@TCSR2    ;SET BIT0 IN TCSR
22796  124532  000005                          RESET                   ;CLEAR BIT0 WITH RESET
22797  124534  032777  000001  177344          BIT     #BIT0,@TCSR2    ;TEST BIT0 CLEAR
22798  124542  001401                          BEQ     TS644
22799  124544  104000                          EMT                     ;
22800
22801
22802                                  ;*********************************************************************************
22803                                  ;TEST 644       TEST THAT BIT6(XMIT INT EN) CAN BE SET & RESET
22804                                  ;*********************************************************************************
22805  124546                          TS644:
22806  124546  017703  177344                  MOV     @TVECT2,R3      ;SAVE XMIT VECTOR
22807  124552  012777  124574  177336          MOV     #1$,@TVECT2     ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
22808  124560  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22809  124564  032777  000100  177314          BIT     #BIT6,@TCSR2    ;TEST BIT6 OF TCSR
```

```
22810  124572  001401                          BEQ     2$
22811  124574                          1$:
22812  124574  104000                          EMT                     ;
22813  124576  052777  000100  177302  2$:     BIS     #BIT6,@TCSR2    ;SET BIT6 OF TCSR
22814  124604  032777  000100  177274          BIT     #BIT6,@TCSR2    ;TEST BIT6 OF TCSR
22815  124612  001001                          BNE     3$
22816  124614  104000                          EMT                     ;
22817  124616  042777  000100  177262  3$:     BIC     #BIT6,@TCSR2    ;CLEAR BIT6 OF TCSR
22818  124624  032777  000100  177254          BIT     #BIT6,@TCSR2    ;TEST BIT6 OF TCSR
22819  124632  001401                          BEQ     4$
22820  124634  104000                          EMT                     ;
22821  124636  032737  000001  001020  4$:     BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
22822  124644  001403                          BEQ     70$             ;IF NO THEN DO TEST
22823  124646  005737  001006                  TST     @#$PASS         ;IF THIS FIRST PASS
22824  124652  001011                          BNE     5$              ;IF NO THEN SKIP TO END OF TEST
22825  124654                          70$:
22826  124654  052777  000100  177224          BIS     #BIT6,@TCSR2    ;SET BIT6 OF TCSR
22827  124662  000005                          RESET                   ;CLEAR BIT6 WITH RESET
22828  124664  032777  000100  177214          BIT     #BIT6,@TCSR2    ;TEST BIT6 OF TCSR
22829  124672  001401                          BEQ     5$
22830  124674  104000                          EMT                     ;
22831  124676  010377  177214          5$:     MOV     R3,@TVECT2      ;RESTORE XMIT VECTOR
22832
22833
22834                                  ;*******************************************************************************
22835                                  ;TEST 645       TEST THAT BIT6 OF RCSR2 CAN BE SET & RESET
22836                                  ;*******************************************************************************
22837  124702                          TS645:
22838  124702  017703  177204                  MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
22839  124706  012777  124730  177176          MOV     #1$,@RVECT2     ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
22840  124714  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22841  124720  032777  000100  177154          BIT     #BIT6,@RCSR2    ;TEST BIT6 OF RCSR
22842  124726  001401                          BEQ     2$
22843  124730                          1$:
22844  124730  104000                          EMT                     ;
22845  124732  052777  000100  177142  2$:     BIS     #BIT6,@RCSR2    ;SET BIT6 OF RCSR
22846  124740  032777  000100  177134          BIT     #BIT6,@RCSR2    ;TEST BIT6 OF RCSR      7
22847  124746  001001                          BNE     3$
22848  124750  104000                          EMT                     ;
22849  124752  042777  000100  177122  3$:     BIC     #BIT6,@RCSR2    ;CLEAR BIT6 OF RCSR
22850  124760  032777  000100  177114          BIT     #BIT6,@RCSR2    ;TEST BIT6 OF RCSR
22851  124766  001401                          BEQ     4$
22852  124770  104000                          EMT                     ;
22853  124772  032737  000001  001020  4$:     BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
22854  125000  001403                          BEQ     70$             ;IF NO THEN DO TEST
22855  125002  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
22856  125006  001011                          BNE     5$              ;IF NO THEN SKIP TO END OF TEST
22857  125010                          70$:
22858  125010  052777  000100  177064          BIS     #BIT6,@RCSR2    ;SET BIT6 OF RCSR
22859  125016  000005                          RESET                   ;CLEAR BIT6OF RCSR2 WITH RESET
22860  125020  032777  000100  177054          BIT     #BIT6,@RCSR2    ;TEST BIT6 OF RCSR
22861  125026  001401                          BEQ     5$
22862  125030  104000                          EMT                     ;
22863  125032  010377  177054          5$:     MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR
22864
22865
```

```
22866
22867                                   ;*****************************************************************************
22868                                   ;TEST 646       TEST THAT XMIT INTERRUPTS ONLY WHEN ENABLED
22869                                   ;*****************************************************************************
22870   125036                          TS646:
22871   125036  042777  000100  177042          BIC     #BIT6,@TCSR2    ;CLEAR TRANSMIT INTERRUPT ENABLE
22872   125044  017703  177046                  MOV     @TVECT2,R3      ;SAVE XMIT VECTOR
22873   125050  012777  125072  177040          MOV     #2$,@TVECT2     ;POINT XMIT VECTOR TO ERROR REPORT
22874   125056  105777  177024          1$:     TSTB    @TCSR2          ;WAIT FOR DONE
22875   125062  100375                          BPL     1$
22876   125064  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22877   125070  000401                          BR      3$
22878   125072                          2$:
22879   125072  104000                          EMT                     ;
22880   125074  012777  125114  177014  3$:     MOV     #4$,@TVECT2     ;SET XMIT VECTOR TO END OF TEST
22881   125102  052777  000100  176776          BIS     #BIT6,@TCSR2    ;ENABLE INTERRUPTS
22882   125110  000240                          NOP
22883
22884   125112  104000                          EMT                     ;XMIT DID NOT INTERRUPT
22885
22886   125114  042777  000100  176764  4$:     BIC     #BIT6,@TCSR2    ;DISABLE INTERRUPTS
22887   125122  022626                          CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22888   125124  010377  176766                  MOV     R3,@TVECT2      ;RESTORE XMIT VECTOR
22889
22890
22891                                   ;*****************************************************************************
22892                                   ;TEST 647       TEST THAT XMIT INTERRUPTS DO NOT OCCUR WHEN DISABLED
22893                                   ;*****************************************************************************
22894   125130                          TS647:
22895   125130  042777  000100  176750          BIC     #BIT6,@TCSR2    ;DISABLE INTERRUPTS
22896   125136  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
22897   125142  017703  176750                  MOV     @TVECT2,R3      ;SAVE XMIT VECTOR
22898   125146  012777  125174  176742          MOV     #2$,@TVECT2     ;POINT XMIT VECTOR TO ERROR REPORT
22899   125154  105777  176726          1$:     TSTB    @TCSR2          ;WAIT FOR DONE
22900   125160  100375                          BPL     1$
22901   125162  052777  000100  176716          BIS     #BIT6,@TCSR2    ;ENABLE INTERRUPT
22902   125170  000240                          NOP
22903   125172  000401                          BR      3$
22904   125174                          2$:
22905   125174  104000                          EMT                     ;
22906   125176  042777  000100  176702  3$:     BIC     #BIT6,@TCSR2    ;CLEAR INTERRUPT ENABLE
22907   125204  012777  125222  176704          MOV     #4$,@TVECT2     ;POINT XMIT VECTOR TO ERROR REPORT
22908   125212  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
22909   125216  000240                          NOP
22910   125220  000401                          BR      5$
22911   125222                          4$:
22912   125222  104000                          EMT                     ;
22913   125224  010377  176666          5$:     MOV     R3,@TVECT2      ;RESTORE XMIT VECTOR
22914
22915
22916                                   ;*****************************************************************************
22917                                   ;TEST 650       TEST TRANSMITTER FOR DOUBLE INTERRUPTS
22918                                   ;*****************************************************************************
22919   125230                          TS650:
22920   125230  042777  000100  176650          BIC     #BIT6,@TCSR2    ;CLEAR INTERRUPT ENABLE
22921   125236  017703  176654                  MOV     @TVECT2,R3      ;SAVE XMIT VECTOR
```

```
22922  125242  017704  176652          MOV     @TPSW2,R4       ;SAVE XMIT PSW VECTOR
22923  125246  012777  125306  176642  MOV     #2$,@TVECT2     ;SET UP XMIT VECTOR
22924  125254  012777  000340  176636  MOV     #340,@TPSW2     ;SET PIO 7 AFTER INTERRUPT
22925  125262  106427  000140          MTPS    #140            ;SET PSW TO PRIORITY 3
22926  125266  105777  176614  1$:     TSTB    @TCSR2          ;WAIT FOR DONE
22927  125272  100375                  BPL     1$
22928  125274  052777  000100  176604  BIS     #BIT6,@TCSR2    ;ENABLE INTERRUPTS
22929  125302  000240                  NOP
22930
22931  125304  104000                  EMT                     ;
22932                                                          ;XMIT INTERRUPT DID NOT OCCUR
22933  125306  022626          2$:     CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
22934  125310  012777  125334  176600  MOV     #4$,@TVECT2     ;POINT XMIT VECTOR TO ERROR
22935  125316  106427  000140          MTPS    #140            ;SET PSW TO PRIORITY 3
22936  125322  000240                  NOP                     ;GIVE TIME FOR ANY INTERRUPTS
22937  125324  042777  000100  176554  BIC     #BIT6,@TCSR2    ;DISABLE INTERRUPTS
22938  125332  000401                  BR      5$
22939  125334                  4$:
22940  125334  104000                  EMT                     ;
22941  125336  010377  176554  5$:     MOV     R3,@TVECT2      ;RESTORE XMIT VECTOR
22942  125342  010477  176552          MOV     R4,@TPSW2       ;RESTORE XMIT PSW VECTOR
22943
22944                          ;*******************************************************************************
22945                          ;TEST 651       TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF2
22946                          ;*******************************************************************************
22947  125346                  TS651:
22948  125346  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
22949  125354  001403                  BEQ     70$             ;IF NO THEN DO TEST
22950  125356  005737  001006          TST     @#$PASS         ;IS THIS FIRST PASS
22951  125362  001043                  BNE     TS652           ;IF NO THEN SHIP TO NEXT TEST
22952  125364                  70$:
22953  125364  042777  000100  176514  BIC     #BIT6,@TCSR2    ;DISABLE INTERRUPTS
22954  125372  106427  000340          MTPS    #340            ;SET PSW TO PRIORITY 7
22955  125376  017703  176514          MOV     @TVECT2,R3      ;SAVE XMIT VECTOR
22956  125402  012777  125452  176506  MOV     #2$,@TVECT2     ;POINT XMIT VECTOR TO ERROR
22957  125410  052777  000100  176470  BIS     #BIT6,@TCSR2    ;ENABLE INTERRUPTS
22958  125416  005077  176466          CLR     @TBUF2          ;LOAD TBUF
22959  125422  105777  176460  1$:     TSTB    @TCSR2          ;WAIT FOR DONE (INTERRUPT)
22960  125426  100375                  BPL     1$
22961  125430  005077  176454          CLR     @TBUF2          ;FILL SECOND BUFFER TO RESET INT.
22962  125434  106427  000140          MTPS    #140            ;SET PSW TO PRIORITY 3
22963  125440  000240                  NOP                     ;GIVE TIME FOR ANY INTERRUPTS
22964  125442  042777  000100  176436  BIC     #BIT6,@TCSR2    ;DISABLE INTERRUPTS
22965  125450  000401                  BR      3$
22966  125452                  2$:
22967  125452  104000                  EMT                     ;
22968  125454  010377  176436  3$:     MOV     R3,@TVECT2      ;RESTORE XMIT VECTOR
22969  125460  005000                  CLR     R0              ;INIT LOOP COUNTER
22970  125462  005200          4$:     INC     R0              ;INCREMENT COUNTER
22971  125464  001376                  BNE     4$              ;UNTIL COUNTER = 0
22972  125466  005777  176412          TST     @RBUF2          ;CLEAR RECEIVER BUFFER
22973
22974
22975                          ;*******************************************************************************
22976                          ;TEST 652       TEST THAT RCVR DONE (7) SET & CLEAR PROPERLY
22977                          ;*******************************************************************************
```

```
22978  125472
22979  125472  005000                  TS652:
22980  125474  005077  176410                  CLR     R0              ;CLEAR A TIMER
22981  125500  105777  176376                  CLR     @TBUF2          ;LOAD TRANSMIT BUFFER
22982  125504  100403          WDONE2: TSTB    @RCSR2          ;CHECK FOR RECEIVER DONE
22983  125506  005200                  BMI     6$              ;BR, IF DONE
22984  125510  001373                  INC     R0              ;INCREMENT TIMER, IF NOT DONE
22985  125512  104000                  BNE     WDONE2
22986                                   EMT                     ;RECEIVER DONE NEVER SET
22987  125514  032737  000001  001020  6$:     BIT     #1,@#$ENV       ;ARE WE RUNNING UNDER APT
22988  125522  001403                  BEQ     70$             ;IF NO THEN DO TEST
22989  125524  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
22990  125530  001005                  BNE     2$              ;IF NO THEN SKIP TO END OF TEST
22991  125532                  70$:
22992  125532  000005                  RESET                   ;CLEAR DONE WITH RESET
22993  125534  105777  176342                  TSTB    @RCSR2          ;CHECK FOR DONE CLEAR
22994  125540  001401                  BEQ     2$
22995  125542  104000                  EMT                     ;RESET DID NOT CLEAR RCVR DONE
22996  125544  005000          2$:     CLR     R0              ;INIT LOOP COUNTER
22997  125546  005200          3$:     INC     R0              ;INCREMENT COUNTER
22998  125550  001376                  BNE     3$              ;UNTIL COUNTER = 0
22999  125552  005777  176326                  TST     @RBUF2          ;CLEAR RECEIVER BUFFER
23000
23001
23002                           ;*********************************************************************************
23003                           ;TEST 653       TEST THAT READING RBUF2 CLEARS RECEIVER DONE
23004                           ;*********************************************************************************
23005  125556                  TS653:
23006  125556  005077  176326                  CLR     @TBUF2          ;LOAD TRANSMITTER
23007  125562  105777  176314          1$:     TSTB    @RCSR2          ;WAIT FOR RECEIVER DONE
23008  125566  100375                  BPL     1$
23009  125570  017700  176310                  MOV     @RBUF2,R0       ;READ RECEIVE BUFFER
23010  125574  105777  176302                  TSTB    @RCSR2          ;CHECK FOR RECEIVE DONE CLEAR
23011  125600  001401                  BEQ     TS654
23012  125602  104000                  EMT                     ;
23013                                                           ;READING RBUF2 DID NOT CLEAR RCVR DONE
23014
23015
23016                           ;*********************************************************************************
23017                           ;TEST 654       TEST THAT RCVR INTERRUPTS ONLY WHEN ENABLED
23018                           ;*********************************************************************************
23019  125604                  TS654:
23020  125604  042777  000100  176274          BIC     #BIT6,@TCSR2    ;DISABLE TRANSMIT INTERRUPTS
23021  125612  042777  000100  176262          BIC     #BIT6,@RCSR2    ;DISABLE RECEIVER INTERRUPTS
23022  125620  017703  176266                  MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
23023  125624  012777  125652  176260          MOV     #2$,@RVECT2     ;POINT RCV VECTOR TO ERROR REPORT
23024  125632  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
23025  125636  005077  176246                  CLR     @TBUF2          ;SEND A CHARACTER
23026  125642  105777  176234          1$:     TSTB    @RCSR2          ;WAIT FOR RECEIVER DONE
23027  125646  100375                  BPL     1$
23028  125650  000401                  BR      3$
23029  125652                  2$:
23030  125652  104000                  EMT                     ;
23031  125654  012777  125674  176230  3$:     MOV     #4$,@RVECT2     ;POINT RCV VECTOR TO END OF TEST
23032  125662  052777  000100  176212          BIS     #BIT6,@RCSR2    ;ENABLE RCV INTERRUPTS
23033  125670  000240                  NOP                     ;GIVE ANY INTERRUPTS TIME
```

```
23034  125672  104000                          EMT                     ;
23035  125674  042777  000100  176200  4$:     BIC     #BIT6,@RCSR2    ;DISABLE INTERRUPTS
23036  125702  022626                          CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
23037  125704  005777  176174                  TST     @RBUF2          ;CLEAR CHARACTER FROM RECEIVER BUFFER
23038  125710  010377  176176                  MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR
23039
23040
23041                                  ;*************************************************************************
23042                                  ;TEST 655       TEST THAT RCVR INTERRUPTS DO NOT OCCUR WHEN DISABLED
23043                                  ;*************************************************************************
23044  125714                          TS655:
23045  125714  106427  000340                  MTPS    #340            ;SET PSW TO PRIORITY 7
23046  125720  017703  176166                  MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
23047  125724  012777  125756  176160          MOV     #2$,@RVECT2     ;POINT RCVR VECTOR TO ERROR REPORT
23048  125732  005077  176152                  CLR     @TBUF2          ;SEND A CHARACTER
23049  125736  105777  176140          1$:     TSTB    @RCSR2          ;WAIT FOR RECEIVER DONE
23050  125742  100375                          BPL     1$
23051  125744  052777  000100  176130          BIS     #BIT6,@RCSR2    ;ENABLE INTERRUPTS
23052  125752  000240                          NOP                     ;GIVE TIME FOR INTERRUPT
23053  125754  000401                          BR      3$
23054  125756                          2$:
23055  125756  104000                          EMT                     ;RCVR INTERRUPTS AT PRIORITY 7
23056  125760  042777  000100  176114  3$:     BIC     #BIT6,@RCSR2    ;CLEAR INTERRUPT ENABLE
23057  125766  012777  126004  176116          MOV     #4$,@RVECT2     ;POINT RCVR VECTOR TO ERROR REPORT
23058  125774  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
23059  126000  000240                          NOP                     ;GIVE TIME FOR ANY INTERRUPT
23060  126002  000401                          BR      5$
23061  126004                          4$:
23062  126004  104000                          EMT                     ;RCVR INTERRUPT REQUEST PASSED WITH BIT6 CLEAR
23063  126006  005777  176072          5$:     TST     @RBUF2          ;CLEAR CHARACTER FROM RECEIVER BUFFER
23064  126012  010377  176074                  MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR
23065
23066
23067                                  ;*************************************************************************
23068                                  ;TEST 656       TEST RECEIVER FOR DOUBLE INTERRUPTS
23069                                  ;*************************************************************************
23070  126016                          TS656:
23071  126016  017703  176070                  MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
23072  126022  017704  176066                  MOV     @RPSW2,R4       ;SAVE RECEIVE PSW VECTOR
23073  126026  012777  126072  176056          MOV     #2$,@RVECT2     ;POINT RCV VECTOR TO CONTINUE TEST
23074  126034  012777  000340  176052          MOV     #340,@RPSW2     ;SET PRIORITY TO 7 AFTER INTERRUPT
23075  126042  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
23076  126046  005077  176036                  CLR     @TBUF2          ;SEND A CHARACTER
23077  126052  105777  176024          1$:     TSTB    @RCSR2          ;WAIT FOR RCVR DONE
23078  126056  100375                          BPL     1$
23079  126060  052777  000100  176014          BIS     #BIT6,@RCSR2    ;ENABLE RCV INTERRUPTS
23080  126066  000240                          NOP                     ;GIVE SOME TIME
23081  126070  104000                          EMT                     ;
23082  126072  022626                  2$:     CMP     (SP)+,(SP)+     ;RESTORE SP AFTER INTERRUPT
23083  126074  012777  126130  176010          MOV     #3$,@RVECT2     ;POINT RCV VECTOR TO ERROR REPORT
23084  126102  106427  000140                  MTPS    #140            ;SET PSW TO PRIORITY 3
23085  126106  000240                          NOP                     ;GIVE SOME TIME
23086  126110  042777  000100  175764          BIC     #BIT6,@RCSR2    ;CLEAR INTERRUPT ENABLE
23087  126116  010377  175770                  MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR
23088  126122  010477  175766                  MOV     R4,@RPSW2       ;RESTORE RECEIVE PSW VECTOR
23089  126126  000401                          BR      4$
```

```
23090  126130
23091  126130  104000                          3$:     EMT             ;
23092  126132  005777  175746                  4$:     TST     @RBUF2          ;CLEAR CHARACTER FROM RECEIVER BUFFER
23093  126136  010377  175750                          MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR
23094
23095                                          ;*****************************************************************************
23096                                          ;TEST 657       TEST THAT RCVR INTERRUPT CLEARS BY READING RBUF2
23097                                          ;*****************************************************************************
23098                                          TS657:
23099  126142
23100  126142  106427  000340                          MTPS    #340            ;SET PSW TO PRIORITY 7
23101  126146  017703  175740                          MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
23102  126152  012777  126222  175732                  MOV     #2$,@RVECT2     ;POINT RCV VECTOR TO ERROR REPORT
23103  126160  052777  000100  175714                  BIS     #BIT6,@RCSR2    ;SET RCVR INTERRUPT ENABLE
23104  126166  005077  175716                          CLR     @TBUF2          ;SEND A CHARACTER
23105  126172  105777  175704                  1$:     TSTB    @RCSR2          ;WAIT FOR DONE (INTERRUPT)
23106  126176  100375                                  BPL     1$
23107  126200  005777  175700                          TST     @RBUF2          ;READ RBUF TO CLEAR PENDING INTERRPUT
23108  126204  106427  000140                          MTPS    #140            ;SET PSW TO PRIORITY 3
23109  126210  000240                                  NOP                     ;ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
23110  126212  042777  000100  175662                  BIC     #BIT6,@RCSR2    ;NO INTERRUPT--CLEAR INT. ENABLE
23111  126220  000401                                  BR      3$
23112  126222                                  2$:
23113  126222  104000                                  EMT             ;
23114  126224  010377  175662                  3$:     MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR
23115
23116
23117
23118                                          ;*****************************************************************************
23119                                          ;TEST 660       TEST THAT RESET CLEARS RECEIVE INTERRUPT
23120                                          ;*****************************************************************************
23121  126230                                  TS660:
23122  126230  032737  000001  001020                  BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
23123  126236  001403                                  BEQ     70$             ;IF NO THEN DO TEST
23124  126240  005737  001006                          TST     @#$PASS         ;IS THIS FIRST PASS
23125  126244  001036                                  BNE     TS661           ;IF NO THEN SHIP TO NEXT TEST
23126  126246                                  70$:
23127  126246  106427  000340                          MTPS    #340            ;SET PSW TO PRIORITY 7
23128  126252  017703  175634                          MOV     @RVECT2,R3      ;SAVE RECEIVE VECTOR
23129  126256  012777  126334  175626                  MOV     #2$,@RVECT2     ;POINT RCV VECTOR TO ERROR REPORT
23130  126264  052777  000100  175610                  BIS     #BIT6,@RCSR2    ;SET RCV INTERRUPT ENABLE
23131  126272  012777  000377  175610                  MOV     #377,@TBUF2     ;SEND AN ALL 1'S CHARACTER
23132  126300  105777  175576                  1$:     TSTB    @RCSR2          ;WAIT FOR RCV DONE
23133  126304  100375                                  BPL     1$
23134  126306  000005                                  RESET                   ;CLEAR RCV INTERRUPT & RBUF2
23135  126310  052777  000100  175564                  BIS     #BIT6,@RCSR2    ;SET RECEIVER INTERRUPT
23136  126316  106427  000140                          MTPS    #140            ;SET PSW TO PRIORITY 3
23137  126322  000240                                  NOP                     ;ALLOW TIME FOR AN ERRONEOUS INTERRUPT
23138  126324  042777  000100  175550                  BIC     #BIT6,@RCSR2    ;NO INTERRUPT-CLEAR INT. ENABLE
23139  126332  000401                                  BR      3$
23140  126334                                  2$:
23141  126334  104000                                  EMT             ;
23142  126336  010377  175550                  3$:     MOV     R3,@RVECT2      ;RESTORE RECEIVE VECTOR
23143
23144
23145                                          ;*****************************************************************************
```

```
23146
23147
23148  126342
23149  126342  012700  000003                ;TEST 661       TEST THAT THE ''OR'' ERROR (BIT14) & ''ERROR'' (BIT15) CAN BE SET
23150  126346  005077  175536                ;*****************************************************************************
23151  126352  105777  175530                TS661:
23152  126356  100375                              MOV     #3,R0           ;SET CHARACTER COUNT TO SEND 3 CHAR.
23153  126360  005300                        1$:   CLR     @TBUF2          ;LOAD TRANSMIT BUFFER
23154  126362  001371                        2$:   TSTB    @TCSR2          ;WAIT FOR TRANSMIT DONE
23155  126364  032777  040000  175512            BPL     2$
23156  126372  001001                            DEC     R0              ;DECREMENT CHARACTER COUNT
23157  126374  104000                            BNE     1$              ;BR IF ALL CHARACTERS NOT TRANSMITTED
23158  126376  032777  100000  175500            BIT     #BIT14,@RBUF2   ;TEST FOR ''OR'' ERROR FLAG
23159  126404  001001                            BNE     3$
23160  126406  104000                            EMT                     ;
23161  126410  005000                        3$:   BIT     #BIT15,@RBUF2   ;TEST ''ERROR'' FLAG
23162  126412  005200                            BNE     4$
23163  126414  001376                            EMT                     ;
23164  126416  005777  175462                4$:   CLR     R0              ;CLEAR LOOP COUNTER
23165                                         5$:   INC     R0              ;INCREMENT LOOP COUNTER
23166                                               BNE     5$              ;  UNTIL COUNTER = 0
23167                                               TST     @RBUF2          ;CLEAR CHARACTER FROM RECEIVER BUFFER
23168
23169                                   ;*****************************************************************************
23170  126422                          ;TEST 662       TEST THAT BREAK TRANSMITS ALL ZEROES
23171  126422  032737  000001  001020  ;*****************************************************************************
23172  126430  001403                  TS662:
23173  126432  005737  001006                BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
23174  126436  001027                        BEQ     70$             ;IF NO THEN DO TEST
23175  126440                                TST     @#$PASS         ;IS THIS FIRST PASS
23176  126440  012777  177777  175442        BNE     TS663           ;IF NO THEN SHIP TO NEXT TEST
23177  126446  105777  175430          70$:
23178  126452  100375                        MOV     #-1,@TBUF2      ;TRANSMIT ALL ONES TO RCVR
23179  126454  005777  175424          1$:   TSTB    @RCSR2          ;WAIT FOR RCVR DONE
23180  126460  052777  000001  175420        BPL     1$
23181  126466  005000                        TST     @RBUF2          ;CLEAR DONE (LEAVING ALL ONES IN RBUF)
23182  126470  105777  175406          2$:   BIS     #BIT0,@TCSR2    ;TRANSMIT BREAK
23183  126474  100403                        CLR     R0              ;CLEAR A TIMER
23184  126476  005200                        TSTB    @RCSR2          ;WAIT FOR RCVR DONE
23185  126500  001373                        BMI     CONT42          ;BR IF DONE
23186  126502  104000                        INC     R0              ;IF NOT, INCREMENT TIMER
23187                                         BNE     2$
23188  126504  105777  175374          CONT42: TSTB  @RBUF2          ;BREAK DID NOT TRANSMIT ANYTHING
23189  126510  001401                        BEQ     3$              ;CHECK RECEIVE BUFFER FOR ZERO
23190  126512  104000                        EMT                     ;BREAK DID NOT TRANSMIT ALL ZEROES
23191
23192  126514  000005                  3$:   RESET                   ;CLEAR ERRORS
23193
23194
23195                                   ;*****************************************************************************
23196                                   ;TEST 663       TEST THAT ''FR'' ERROR CAN BE SET DURING BREAK
23197                                   ;*****************************************************************************
23198  126516                          TS663:
23199  126516  052777  000001  175362        BIS     #BIT0,@TCSR2    ;SEND BREAK
23200  126524  005077  175360                CLR     @TBUF2          ;TRANSMIT A CHARACTER TO TIME BREAK
23201  126530  105777  175346          1$:   TSTB    @RCSR2          ;WAIT FOR RCVR DONE
```

```
23202  126534  100375                          BPL     1$
23203  126536  042777  000001  175342          BIC     #BIT0,@TCSR2    ;CLEAR BREAK BITS
23204  126544  032777  020000  175332          BIT     #BIT13,@RBUF2   ;CHECK FOR FRAMING ERROR FLAG
23205  126552  001001                          BNE     2$
23206  126554  104000                          EMT                     ;BREAK DID NOT SET FRAMING ERROR
23207  126556  032777  100000  175320  2$:     BIT     #BIT15,@RBUF2   ;TEST ''ERROR'' FLAG
23208  126564  001001                          BNE     3$
23209  126566  104000                          EMT                     ;''ERROR'' FLAG DID NOT SET WITH ''OR'' FLAG
23210  126570  005777  175310          3$:     TST     @RBUF2          ;CLEAR RECEIVER BUFFER
23211
23212                                  ;*********************************************************************************************
23213                                  ;TEST 664       TEST DATA PATHS USING WRAP CABLE
23214                                  ;*********************************************************************************************
23215  126574                          TS664:
23216  126574  005001                          CLR     R1              ;CLEAR REGISTER FOR TEST DATA
23217  126576  105201                  1$:     INCB    R1              ;INCREMENT THE TEST DATA
23218  126600  010177  175304                  MOV     R1,@TBUF2       ;XMIT A CHARACTER
23219  126604  005000                          CLR     R0              ;CLEAR A TIMER
23220  126606  105777  175270          2$:     TSTB    @RCSR2          ;WAIT FOR RECEIVER DONE
23221  126612  100403                          BMI     3$              ;BR IF DONE
23222  126614  005200                          INC     R0              ;INCREMENT TIMER IF NOT
23223  126616  001373                          BNE     2$
23224  126620  104000                          EMT                     ;
23225  126622  017702  175256          3$:     MOV     @RBUF2,R2       ;GET RECEIVED CHARACTER
23226  126626  020102                          CMP     R1,R2           ;COMPARE DATA
23227  126630  001401                          BEQ     4$
23228  126632  104000                          EMT                     ;
23229  126634  105701                  4$:     TSTB    R1              ;TEST XMIT DATA FOR ZERO
23230  126636  001357                          BNE     1$              ;BR, IF NOT FINISHED
23231  126640  000167  000050                  JMP     UNIQUE          ;FINISHED TESTING DEVICES SEPARATELY
23232                                                                  ;GO TEST THEM ALL TOGETHER
23233  126644  000000                  $BDADR: 0
23234  126646  000000                  $BDDAT: 0
23235  126650  000000                  $GDADR: 0
23236  126652  000000                  $GDDAT: 0
23237
23238
23239  126654  012737  000006  001002  ERROR7: MOV     #6,@#$FATAL             ;SET UP FATAL ERROR NUMBER
23240  126662  012767  000001  052110          MOV     #1,$MSGTY               ;SET FATAL ERROR FLAG
23241  126670  000777                  SL2HLT: BR      .
23242
23243
23244  126672  177560                  DADTBL: .WORD   177560
23245  126674  177562                          .WORD   177562
23246  126676  177564                          .WORD   177564
23247  126700  177566                          .WORD   177566
23248  126702  176500                          .WORD   176500
23249  126704  176502                          .WORD   176502
23250  126706  176504                          .WORD   176504
23251  126710  176506                          .WORD   176506
23252  126712  177564                  TBLEND: .WORD   177564
23253
23254
23255  126714  032777  000034  072342  UNIQUE: BIT     #34,@SWR
23256  126722  001402                          BEQ     1$
23257  126724  000167  002060                  JMP     ENDPAS
```

```
23258  126730  012737  000007  001004  1$:     MOV     #7,@#$TESTN     ;UPDATE TEST NUMBER FOR APT
23259  126736  012737  127772  000030          MOV     #ERROR8,@#30    ;SET UP FOR CORRECT ERROR CALL
23260
23261                                  ;*****************************************************************************
23262                                  ;TEST 665       UNIQUE INTERNAL ADDRESS TEST
23263                                  ;*****************************************************************************
23264  126744                          TS665:
23265  126744  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
23266  126752  001403                          BEQ     70$             ;IF NO THEN DO TEST
23267  126754  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
23268  126760  001044                          BNE     TS666           ;IF NO THEN SHIP TO NEXT TEST
23269  126762                          70$:
23270  126762  012767  000340  051006          MOV     #340,PS         ;WE WILL BE PLAYING WITH BIT6
23271                                                                  ;SO LOCK OUT EXTRAINEOUS INTERRUPTS
23272  126770  012700  126672                  MOV     #DADTBL,R0      ;GET LOCATION OF FIRST REGISTER ADDRESS
23273  126774  012703  126672          1$:     MOV     #DADTBL,R3      ;MAKE R3 POINT TO LOCATION OF FIRST
23274                                                                  ;REGISTER ADDRESS
23275  127000  012701  000011                  MOV     #11,R1          ;SET LOOP COUNTER TO CLEAR ALL REG.
23276  127004  005033                  2$:     CLR     @(R3)+          ;CLEAR A REGISTER
23277  127006  077102                          SOB     R1,2$           ;LOOP UNTIL ALL REGISTERS CLEARED
23278  127010  012770  000100  000000          MOV     #BIT6,@(R0)     ;SET TEST BIT IN DEVICE REGISTERS
23279  127016  012701  126672                  MOV     #DADTBL,R1      ;GET LOCATION OF FIRST REGISTER ADDRESS
23280  127022  012702  000011                  MOV     #11,R2          ;SET UP TEST LOOP COUNTER
23281  127026  032731  000100          3$:     BIT     #BIT6,@(R1)+    ;IS TEST BIT SET IN THIS REGISTER
23282  127032  001006                          BNE     5$              ;IF YES GO SEE IF THERE IS AN ERROR
23283  127034  077204                  4$:     SOB     R2,3$           ;LOOP UNTIL ALL REGISTER CHECKED
23284  127036  005030                          CLR     @(R0)+          ;CLEAR REGISTER JUST TESTED AND POINT
23285                                                                  ;TO NEXT ONE
23286  127040  020027  126712                  CMP     R0,#TBLEND              ;ARE WE DONE TESTING
23287  127044  001407                          BEQ     7$              ;IF YES GO TO NEXT TEST
23288  127046  000752                          BR      1$              ;CONTINUE TESTING
23289  127050  021041          5$:     CMP     (R0),-(R1)      ;DID WE COMPARE THE REGISTER TO ITSELF?
23290  127052  001401                          BEQ     6$
23291  127054  104000                          EMT                     ;WRITE TO 1 INTERNAL ADDRESS MODIFIED
23292                                                                  ;ANOTHER SO ADDRESS NOT UNIQUE
23293  127056  062701  000002          6$:     ADD     #2,R1           ;RESTORE POINTER
23294  127062  000764                          BR      4$              ;GET BACK IN TEST LOOP
23295  127064  005000                  7$:     CLR     R0              ;INITIALIZE LOOP COUNTER
23296  127066  005200                  8$:     INC     R0              ;INCREMEMT COUNTER
23297  127070  001376                          BNE     8$              ;  UNTIL LOOP COUNTER = 0
23298
23299
23300
23301                                  ;*****************************************************************************
23302                                  ;TEST 666       TEST ALL INTERNAL OPTIONS SIMULTANEOUSLY
23303                                  ;*****************************************************************************
23304  127072                          TS666:
23305  127072  032737  000001  001020          BIT     #1, @#$ENV      ;ARE WE RUNNING UNDER APT
23306  127100  001405                          BEQ     70$             ;IF NO DO TEST
23307  127102  005737  001006                  TST     @#$PASS         ;IS THIS FIRST PASS
23308  127106  001402                          BEQ     70$             ;IF YES DO TEST
23309  127110  000167  001674                  JMP     ENDPAS          ;IF NO THEN SKIP THIS TEST
23310  127114  000005          70$:    RESET                   ;CLEAR EVERY BODY
23311  127116  012767  000340  050652          MOV     #340,PS         ;SET PROCESSOR PRIORITY TO 7
23312  127124  017767  174766  110106          MOV     @TVECT2,$TMP0
23313  127132  017767  174754  110102          MOV     @RVECT2,$TMP1
```

```
23314  127140  017767  171106  170076          MOV     @TVECT,$TMP2
23315  127146  017767  171074  170072          MOV     @RVECT,$TMP3
23316  127154  017767  171100  170066          MOV     @RTCVT,$TMP4
23317  127162  005067  000572                  CLR     XMTCT1          ;INITIALIZE COUNTERS
23318  127166  005067  000570                  CLR     XMTCT2
23319  127172  005067  000566                  CLR     RECCT1
23320  127176  005067  000564                  CLR     RECCT2
23321  127202  005067  000562                  CLR     TICKS
23322  127206  012777  127406  171036          MOV     #XMIT1,@TVECT   ;SET UP SLU1 TRANSMIT VECTOR
23323  127214  012777  000340  171032          MOV     #340, @TPSW     ;AND PSW
23324  127222  012777  127442  171016          MOV     #REC1, @RVECT   ;SET UP SLU1 RECEIVER VECTOR
23325  127230  012777  000340  171012          MOV     #340, @RPSW     ;AND PSW
23326  127236  012777  127470  174652          MOV     #XMIT2, @TVECT2 ;SET UP SLU2 TRANSMIT VECTOR
23327  127244  012777  000340  174646          MOV     #340, @TPSW2    ;AND PSW
23328  127252  012777  127524  174632          MOV     #REC2, @RVECT2  ;SET UP SLU2 RECEIVER VECTOR
23329  127260  012777  000340  174626          MOV     #340, @RPSW2    ;AND PSW
23330  127266  012777  127376  170764          MOV     #TICKER, @RTCVT ;SET UP RTC VECTOR
23331  127274  012777  000340  170760          MOV     #340, @RTCPSW   ;AND PSW
23332  127302  052777  000004  170732          BIS     #BIT2, @TCSR    ;ENABLE SLU1 MAINTENANCE WRAP
23333  127310  052777  000100  170724          BIS     #BIT6, @TCSR    ;ENABLE SLU1 XMIT INTERRUPT
23334  127316  052777  000100  170712          BIS     #BIT6, @RCSR    ;ENABLE SLU1 RECEIVER INTERRUPT
23335  127324  012702  130010                  MOV     #BUF1, R2       ;SET UP RECEIVER BUFFER
23336  127330  052777  000100  174550          BIS     #BIT6, @TCSR2   ;ENABLE SLU2 XMIT INTERRUPT
23337  127336  052777  000100  174536          BIS     #BIT6, @RCSR2   ;ENABLE SLU2 RECEIVER INTERRUPT
23338  127344  012703  130410                  MOV     #BUF2, R3       ;SET UP RECEIVER BUFFER
23339  127350  052777  000100  170700          BIS     #BIT6, @LKS     ;ENABLE RTC INTERRUPTS
23340  127356  012700  177777          3$:     MOV     #-1, R0         ;INITIALIZE DATA FOR SLU1
23341  127362  012701  177777                  MOV     #-1, R1         ;INITIALIZE DATA FOR SLU2
23342  127366  005067  050404                  CLR     PS              ;DROP PROCESSOR PRIORITY TO 0
23343  127372  000001                  WAITIO: WAIT                    ;WAIT FOR INTERRUPT
23344  127374  000776                  BR      WAITIO
23345
23346  127376  005267  000366          TICKER: INC     TICKS           ;UPDATE COUNT
23347  127402  000167  000140                  JMP     IOHAND          ;GO TO INTERRUPT HANDLER
23348  127406  005267  000346          XMIT1:  INC     XMTCT1          ;UPDATE XMIT INTERRUPT COUNT
23349  127412  005200                  INC     R0              ;UPDATE XMIT DATA
23350  127414  010077  170624                  MOV     R0, @TBUF       ;SEND NEXT CHARACTER
23351  127420  026727  000334  000400          CMP     XMTCT1,#400     ;IF 256 CHARACTERS HAVE NOT BEEN
23352  127426  002403                  BLT     1$              ;TRANSFERRED CONTINUE
23353  127430  042777  000100  170604          BIC     #100,   @TCSR   ;ELSE NO MORE TRANSMITTER INTERRUPTS
23354  127436  000167  000104          1$:     JMP     IOHAND          ;GO TO INTERRUPT HANDLER
23355
23356  127442  005267  000316          REC1:   INC     RECCT1          ;UPDATE RECEIVER INTERUPT COUNT
23357  127446  005777  170566                  TST     @RBUF           ;BIT 15 SET IF ANY ERRORS OCCURRED
23358  127452  100002                  BPL     3$              ;IF BIT IS CLEAR NO ERROR
23359  127454  000005                  RESET                   ;CLEAR THE WORLD STOP ALL
23360                                                          ;INTERRUPTS
23361  127456  104000                  EMT                     ;RECEIVER STATUS ERROR
23362  127460  117722  170554          3$:     MOVB    @RBUF, (R2)+    ;GET DATA AND STORE IT
23363  127464  000167  000056                  JMP     IOHAND          ;GO TO INTERRUPT HANDLER
23364
23365  127470  005267  000266          XMIT2:  INC     XMTCT2          ;UPDATE XMIT INTERRUPT COUNT
23366  127474  005201                  INC     R1              ;UPDATE XMIT DATA
23367  127476  010177  174406                  MOV     R1, @TBUF2      ;SEND NEXT CHARACTER
23368  127502  026727  000254  000400          CMP     XMTCT2, #400    ;IF 256 CHARACTERS HAVE NOT
23369  127510  002403                  BLT     1$              ;BEEN TRANSFERRED CONTINUE
```

```
23370  127512  042777  000100  174366          BIC     #BIT6, @TCSR2   ;ELSE NO MORE XMIT INTERRUPTS
23371  127520  000167  000022          1$:     JMP     IOHAND          ;GO TO INTERRUPT HANDLER
23372
23373  127524  005267  000236          REC2:   INC     RECCT2          ;UPDATE RECEIVER INTERRUPT COUNT
23374  127530  005777  174350                  TST     @RBUF2          ;BIT 15 SETS IF ANY ERRORS OCCURRED
23375  127534  100002                          BPL     3$              ;IF BIT IS CLEAR NO ERRORS
23376  127536  000005                          RESET                   ;CLEAR THE WORLD - STOP ALL
23377                                                                  ;INTERRUPTS
23378  127540  104000                          EMT                     ;RECEIVER STATUS ERROR
23379  127542  117723  174336          3$:     MOVB    @RBUF2, (R3)+   ;GET DATA AND STORE IT
23380
23381  127546  026727  000216  000074  IOHAND: CMP     TICKS, #74      ;HAS 1 SEC ELAPSED
23382  127554  001401                          BEQ     1$              ;IF YES STOP TEST
23383  127556  000002                          RTI                     ;RETURN FROM INTERRUPT TO AWAIT NEXT
23384  127560  042777  000100  170454  1$:     BIC     #BIT6, @TCSR    ;IF YES STOP TRANSMISSIONS
23385  127566  042777  000100  174312          BIC     #BIT6, @TCSR2   ;
23386  127574  042777  000100  170454          BIC     #BIT6,  @LKS    ;TURN OFF LINE CLOCK
23387
23388  127602  106427  000000          WAITER: MTPS    #0              ;LOWER PRIORITY TO ALLOW TIME FOR RECEIVER TO FINISH
23389  127606  012705  140000                  MOV     #-40000,R5      ;SET UP LOOP COUNTER
23390  127612  062705  000001          1$:     ADD     #1, R5          ;DO LOOP UNTIL R5 = 0
23391  127616  001375                          BNE     1$
23392  127620  000005                          RESET                   ;STOP EVERYONE SHOULD BE DONE
23393
23394  127622  026767  000132  000134  CHECK1: CMP     XMTCT1,RECCT1   ;# OF XMIT INTERRUPTS = REC INTERRUPTS
23395  127630  001401                          BEQ     1$
23396  127632  104000                          EMT                     ;INTERRUPT COMPARISON ERROR
23397  127634  012702  130010          1$:     MOV     #BUF1, R2       ;POINT TO FIRST DATA
23398  127640  005000                          CLR     R0              ;INITIALIZE TO FIRST DATA XMIT
23399  127642  016704  000112                  MOV     XMTCT1, R4      ;GET # OF BYTES TRANSFERRED
23400  127646  122200          2$:     CMPB    (R2)+, R0       ;IS RECEIVED DATA = EXPECTED
23401  127650  001401                          BEQ     3$
23402  127652  104000                          EMT                     ;SLU1 DATA COMPARISON ERROR
23403  127654  005200          3$:     INC     R0              ;UPDATE TO NEXT GOOD DATA
23404  127656  077405                          SOB     R4,2$           ;LOOP UNTIL ALL DATA CHECKED
23405  127660  026767  000076  000100  CHECK2: CMP     XMTCT2, RECCT2  ;#OF XMIT INTERRUPTS = REC INTERRUPTS
23406  127666  001401                          BEQ     1$
23407  127670  104000                          EMT                     ;INTERRUPT COMPARISON ERROR
23408  127672  012703  130410          1$:     MOV     #BUF2, R3       ;INITIALIZE TO FIRST RECEIVED DATA
23409  127676  005001                          CLR     R1              ;INITIALIZE TO FIRST XMIT DATA
23410  127700  016704  000056                  MOV     XMTCT2, R4      ;GET # OF BYTES TRANSFERRED
23411  127704  122301          2$:     CMPB    (R3)+, R1       ;IS RECEIVED DATA = EXPECTED DATA
23412  127706  001401                          BEQ     3$
23413  127710  104000                          EMT                     ;SLU2 DATA COMPARISON ERROR
23414  127712  005201          3$:     INC     R1              ;UPDATE TO NEXT GOOD DATA
23415  127714  077405                          SOB     R4,2$           ;LOOP UNTIL ALL DATA CHECKED
23416  127716  016777  107316  174172  FINIE:  MOV     $TMP0, @TVECT2  ;RESTORE VECTORS
23417  127724  016777  107312  174160          MOV     $TMP1, @RVECT2
23418  127732  016777  107306  170312          MOV     $TMP2, @TVECT
23419  127740  016777  107302  170300          MOV     $TMP3, @RVECT
23420  127746  016777  107276  170304          MOV     $TMP4, @RTCVT
23421  127754  000167  001030                  JMP     ENDPAS          ;FINISHED TESTING GO TO END OF PASS
23422
23423  127760  000000          XMTCT1: .WORD   0
23424  127762  000000          XMTCT2: .WORD   0
23425  127764  000000          RECCT1: .WORD   0
```

```
23426  127766  000000                  RECCT2: .WORD   0
23427  127770  000000                  TICKS:  .WORD   0
23428
23429  127772  012737  000007  001002  ERROR8: MOV     #7,@#$FATAL             ;SET UP FATAL ERROR NUMBER
23430  130000  012767  000001  050772          MOV     #1,$MSGTY               ;SET FATAL ERROR FLAG
23431  130006  000777                  COMHLT: BR      .
23432
23433  130010  000200                  BUF1:   .BLKW   200
23434  130410  000200                  BUF2:   .BLKW   200
23435
23436
23437
23438
23439  131010  005327                  ENDPAS: DEC     (PC)+           ;DECREMENT TEST LOOP COUNTER
23440  131012  000001                  $EOPCT: .WORD   1
23441  131014  003043                          BGT     $DOAGN          ;IF COUNTER NOT 0 DO TEST AGAIN
23442  131016  005267  047764                  INC     $PASS           ;INCREMENT PASS COUNTER
23443  131022  042767  100000  047756          BIC     #100000,$PASS   ;DON'T LET IT BE NEGATIVE
23444  131030  016767  047770  177754          MOV     $USWR,$EOPCT    ;RESET TEST LOOP COUNTER
23445  131036  012700  131247                  MOV     #ENDMSG,R0      ;LET R0 POINT TO ENDPASS MESSAGE
23446  131042  004767  000126                  JSR     PC,TYPE         ;GO TYPE END PASS MESSAGE
23447  131046  016700  046770                  MOV     42,R0           ;GET MONITOR ADDRESS
23448  131052  001405                          BEQ     DOAGIN          ;IF = 0 NO MONITOR SO DON'T STOP
23449  131054  000005                          RESET                   ;IF MONITOR CLEAR THE WORLD
23450  131056  004710                  $ENDAD: JSR     PC,(R0)         ;GO TO MONITOR
23451  131060  000240                          NOP                     ;THESE THREE LOCATIONS RESERVED
23452  131062  000240                          NOP
23453  131064  000240                          NOP
23454  131066  013737  000004  037240  DOAGIN: MOV     @#4,@#$TMP0
23455  131074  012737  131112  000004          MOV     #1$,@#4
23456  131102  012737  000001  164000          MOV     #1,@#164000
23457  131110  000402                          BR      2$
23458  131112  062706  000004          1$:     ADD     #4,SP
23459  131116  013737  037240  000004  2$:     MOV     @#$TMP0,@#4
23460  131124  000137                  $DOAGN: JMP     @(PC)+          ;RETURN TO TEST AT LOCATION RESTRT
23461  131126  001206                          .WORD   RESTRT
23462
23463
23464
23465                                   ;*******************************************************************************
23466                                   ;*COMMON SUBROUTINES THAT ARE NEEDED BY THE PROGRAM
23467                                   ;*
23468                                   ;*******************************************************************************
23469
23470  131130  012737  131140  000024  PWRDN:  MOV     #PWRUP,@#24     ;SET UP POWER FAIL VECTOR FOR POWER UP
23471  131136  000000                          HALT
23472
23473  131140  012737  131130  000024  PWRUP:  MOV     #PWRDN,@#24     ;SET UP POWER FAIL VECTOR FOR POWER DOWN
23474  131146  012706  001000                  MOV     #STBOT,SP       ;SET UP STACK
23475  131152  005737  000172                  TST     @#MTFLAG        ;ARE WE ON MULTI-OPTION TESTER
23476  131156  001004                          BNE     1$              ;IF YES SKIP TYPE OUT
23477  131160  012700  131230                  MOV     #PWRMSG,R0      ;POINT R0 TO POWER FAIL MESSAGE
23478  131164  004767  000004                  JSR     PC,TYPE         ;GO TYPE IT
23479  131170  000167  050012          1$:     JMP     RESTRT          ;GO RESTART TEST
23480
23481
```

```
23482  131174  132767  000040  047617  TYPE:   BITB    #40,$ENVM       ;TYPE OUTS DISABLED
23483  131202  001011                  BNE     3$              ;IF YES GET TO EXIT
23484  131204  105737  177564          1$:     TSTB    @#TTCSR         ;TEST FOR PRINTER READY BIT
23485  131210  100375                  BPL     1$              ;IF NOT READY WAIT FOR IT
23486  131212  112037  177566          MOVB    (R0)+,@#TPB     ;WHEN READY PRINT A CHARACTER
23487  131216  001372                  BNE     1$              ;IF LAST CHARACTER NOT NULL CONTINUE TYPING
23488  131220  105737  177564          2$:     TSTB    @#TTCSR         ;WAIT FOR PRINTER TO FINISH
23489  131224  100375                  BPL     2$
23490  131226  000207                  3$:     RTS     PC
23491
23492                                  ;*********************************************************************************************
23493                                  ;*MESSAGES
23494                                  ;*
23495                                  ;*********************************************************************************************
23496
23497  131230  047520  042527  020122  PWRMSG: .ASCIZ  /POWER FAILED/<12><15>
23498  131236  040506  046111  042105
23499  131244  006412     000
23500  131247     105  042116  047440  ENDMSG: .ASCIZ  /END OF PASS CJKDEA0/<12><15>
23501  131254  020106  040520  051523
23502  131262  041440  045512  042504
23503  131270  030101  006412     000
23504
23505
23506          000001                  .END
```

```
A       020330  5726# 5732* 5733  5736  5740  5750* 5751  5754  5758  5768* 5769* 5770* 5771
                5774  5778  5782  5787  5795* 5796* 5797  5798* 5801  5805  5811  5815  5840
                5842  5850* 5851  5853  5861* 5862* 5863  5871* 5872* 5873
AAADON  067716  15329 15392#
AAA1    067116  15164#
AAA10   067446  15274#
AAA11   067476  15288#
AAA12   067526  15302#
AAA13   067556  15316#
AAA2    067146  15174#
AAA3    067176  15185#
AAA4    067226  15196#
AAA5    067256  15208#
AAA6    067306  15219#
AAA7    067336  15233#
AAA8    067366  15247#
AAA9    067416  15261#
AABBF0  103714  18580 18584 18591 18597 18606#
AABDON  103734  18599 18615#
AABTP1  103724  18579 18611#
AAB2    103712  18593 18596 18598 18602#
AACDON  110556  19949 19958#
AACTP1  110550  19937 19945 19953#
AAC10   110554  19940 19946 19948 19955#
AAC2    110524  19939 19941#
AADATO  061400  13764 13783 13805 13814#
AADBF   117674  21758* 21759 21762 21766#
AADDON  117700  21763 21769#
AADONE  061500  13813 13846#
AAD1    117620  21752#
AAPAT0  061410  13758 13778 13796 13818#
AAPAT1  061420  13761 13780 13822#
AAPAT2  061430  13766 13826#
AAPAT3  061440  13785 13830#
AAPAT4  061450  13834#
AAPAT5  061460  13799 13838#
AAPAT6  061470  13807 13842#
AA11    061270  13781#
AA12    061272  13783#
AA13    061310  13787# 13790
AA2     061220  13762#
AA20    061316  13788 13790#
AA22    061342  13800#
AA23    061344  13802#
AA24    061364  13809# 13812
AA27    061372  13810 13812#
AA3     061222  13764#
AA4     061240  13768# 13771
AA7     061246  13769 13771#
ABASE = 000000  557
ACDW1 = 000000  557
ACDW2 = 000000  557
ACPUOP= 000000  557   572
ADC1    014274  4532  4533  4535#
ADC2    014276  4534  4537#
ADC3    014316  4541  4542  4544#
```

```
ADC4    014320          4543    4546#
ADC5    014336          4549    4550    4551    4553#
ADDW0 = 000000           557
ADDW1 = 000000           557
ADDW10= 000000           557
ADDW11= 000000           557
ADDW12= 000000           557
ADDW13= 000000           557
ADDW14= 000000           557
ADDW15= 000000           557
ADDW2 = 000000           557
ADDW3 = 000000           557
ADDW4 = 000000           557
ADDW5 = 000000           557
ADDW6 = 000000           557
ADDW7 = 000000           557
ADDW8 = 000000           557
ADDW9 = 000000           557
ADD1    014156          4487    4488    4490#
ADD2    014160          4489    4492#
ADD3    014174          4495    4496    4498#
ADD4    014176          4497    4500#
ADD5    014214          4503    4504    4506#
ADD6    014216          4505    4508#
ADD7    014230          4509    4510    4512#
ADD8    014232          4511    4514#
ADD9    014252          4517    4518    4519    4521#
ADEVCT= 000000           557     563
ADEVM = 000000           557
ADONE   050452         11532#
AENV  = 000000           557     568
AENVM = 000000           557     569
AERR1   050442         11509   11510   11511   11527#
AFATAL= 000000           557     560
AMADR1= 000000           557
AMADR2= 000000           557
AMADR3= 000000           557
AMADR4= 000000           557
AMAMS1= 000000           557
AMAMS2= 000000           557
AMAMS3= 000000           557
AMAMS4= 000000           557
AMSGAD= 000000           557     565
AMSGLG= 000000           557     566
AMSGTY= 000000           557     559
AMTYP1= 000000           557
AMTYP2= 000000           557
AMTYP3= 000000           557
AMTYP4= 000000           557
APASS = 000000           557     562
APRIOR= 000000           557
APTENV= 000001          542#    6617
AROUN   001514           719     722#
AROUND  024724          6760    6771#
ASLB1   015654          4999    5001#
ASLB2   015642          4993    4994    4996#
```

```
ASLB3   015644          4995    4998#
ASLB4   015666          5002    5003    5004    5006#
ASL1    015246          4849    4850    4851    4853#
ASL2    015250          4852    4855#
ASL3    015266          4858    4859    4860    4862#
ASL4    015270          4861    4864#
ASL5    015304          4867    4868    4870#
ASL6    015306          4869    4872#
ASL7    015332          4875    4876    4877    4878    4881#
ASRB1   015712          5019    5021#
ASRB2   015724          5022    5023    5024    5026#
ASRB3   015726          5025    5028#
ASRB4   015734          5029    5031#
ASRB5   015736          5030    5033#
ASRB6   015756          5036    5037    5038    5040#
ASRB7   015760          5039    5042#
ASR1    015356          4891    4892    4893    4895#
ASR2    015360          4894    4897#
ASR3    015402          4901    4902    4903    4905#
ASR4    015404          4904    4907#
ASR5    015420          4910    4911    4912    4914#
ASR6    015422          4913    4916#
ASR7    015452          4920    4921    4922    4923    4926#
ASTART  025414          6927#   7101    7105
ASWREG= 000000           557     570
ATESTN= 000000           557     561
ATRAP   023340          6450    6469#
AUNIT = 000000           557     564
AUSWR = 000000           557     571
AUTO1   023432          6495#
AVECT1= 000000           557
AVECT2= 000000           557
A1      050412         11515#  11531
A11     050412         11516#
A12     050426         11521#
A2      050444         11526   11530#
BBBDON  070172         15481   15491#
BBBER1  067766         15411   15424#
BBBP1   070152         15412   15414   15437   15460   15484#
BBBP2   070162         15435   15456   15486#
BBB0    067722         15407#
BBB1    067750         15416#
BBB2    070002         15428   15432#
BBB3    070022         15438#
BBB4    070054         15449   15453#
BBB5    070102         15462#  15465
BBB6    070110         15459   15465#
BBB7    067770         15423   15426#
BBCDON  110632         19984   19995#
BBCTP1  110624         19972   19980   19989#
BBC10   110630         19974   19981   19983   19993#
BBC2    110600         19976#
BBDATO  062760         14040   14060   14081   14101   14120   14139   14150#
BBDDON  120130         21819   21828#
BBDONE  063110         14148   14194#
BBD1    117710         21782#
```

```
BBPAT0  062770          14072   14083   14154#
BBPAT1  063000          14037   14057   14077   14097   14117   14136   14158#
BBPAT2  063010          14035   14042   14162#
BBPAT3  063020          14093   14166#
BBPAT4  063030          14055   14170#
BBPAT5  063040          14115   14174#
BBPAT6  063050          14134   14141   14178#
BBP10   063070          14062   14186#
BBP11   063100          14122   14190#
BBP7    063060          14103   14182#
BB10    062436          14064#  14067
BB13    062444          14065   14067#
BB14    062454          14069   14072#
BB15    062474          14078#
BB16    062520          14085#  14088
BB17    062526          14086   14088#
BB2     062334          14038#
BB20    062536          14090   14093#
BB21    062556          14098#
BB22    062602          14105#  14108
BB25    062610          14106   14108#
BB26    062620          14110   14113#
BB27    062640          14118#
BB3     062340          14040#
BB30    062662          14124#  14127
BB31    062670          14125   14127#
BB32    062700          14129   14132#
BB33    062720          14137#
BB34    062742          14143#  14146
BB35    062750          14144   14146#
BB4     062356          14044#  14047
BB5     062364          14045   14047#
BB6     062374          14050   14053#
BB7     062414          14058#
BDONE   050506          11558#
BELL  = 000240            530#
BIC1    013536           4280    4281    4283#
BIC2    013540           4282    4285#
BIC3    013556           4288    4289    4291#
BIS1    013602           4301    4302    4303    4305#
BIS2    013604           4304    4307#
BIS3    013624           4310    4311    4312    4314#
BITCHK  020004           5609#
BITCLR  017732           5595#
BITCON  020060           5622    5625#
BITSET  017750           5602#
BITST1  013472           4258    4259    4261#
BITST2  013474           4260    4264#
BITST3  013512           4267    4268    4270#
BIT0  = 000001           9246#   9817    9841    9865    9890    9911    9941    9965    9989    10013   10055   21857#  22778
                        22781   22782   22785   22786   22795   22797   23180   23199   23203
BIT1  = 000002          21858#
BIT10 = 002000          21867#
BIT11 = 004000          21868#
BIT12 = 010000          21869#
BIT13 = 020000          21870#  23204
```

```
BIT14  = 040000   21871# 22363  23155
BIT15  = 100000   21872# 22366  23158  23207
BIT2   = 000004   21859# 21999  22002  22003  22006  22007  22010  22012  22181  22188  22204  22209  22222
                  22229  22232  22251  22258  22264  22267  22282  22286  22312  22316  22336  22355  22362
                  22381  22389  22393  23332
BIT3   = 000010   21860#
BIT4   = 000020    9864   9910   9940   9964   9988  10012  10054  21861#
BIT5   = 000040   21862#
BIT6   = 000100   21796  21810  21863# 22024  22028  22029  22032  22033  22037  22039  22052  22056  22057
                  22060  22061  22065  22067  22078  22088  22093  22102  22108  22113  22127  22135  22144
                  22155  22159  22166  22220  22221  22235  22238  22255  22260  22287  22294  22311  22320
                  22335  22341  22344  22427  22431  22432  22435  22436  22444  22446  22490  22508  22516
                  22539  22553  22571  22577  22592  22598  22616  22809  22813  22814  22817  22818  22826
                  22828  22841  22845  22846  22849  22850  22858  22860  22871  22881  22886  22895  22901
                  22906  22920  22928  22937  22953  22957  22964  23020  23021  23032  23035  23051  23056
                  23079  23086  23103  23110  23130  23135  23138  23278  23281  23333  23334  23336  23337
                  23339  23370  23384  23385  23386
BIT7   = 000200   21864# 22464  22465  22467  22468  22489  22507  22538  22570  22591  22595
BIT8   = 000400   21865#
BIT9   = 001000   21866#
BRCT     001540     706*   734#  740*
BRH      001524     707*   727#
BRTAB    021122     704   5937#
BR1      021322    6024   6027#
BR10     021464    6070   6073#
BR11     021544    6088   6091#
BR12     021604    6097   6100#
BR13     021644    6106   6109#
BR14     021704    6115   6118#
BR15     021756    6131   6134#
BR16     021770    6135   6138#
BR17     022002    6139   6142#
BR2      021342    6030   6033#
BR20     022014    6143   6145#
BR21     022026    6146   6149#
BR22     022040    6150   6153#
BR23     022052    6154   6157#
BR3      021356    6036   6039#
BR4      021374    6043   6046#
BR5      021412    6050   6053#
BR6      021430    6057   6059#
BR7      021446    6063   6066#
BR70     023160    6414#
BR71     023674    6538   6545#
BTCON    020116    5641   5644#
BTERR    020110    5642#
BTRAP    023406    6477   6489#
BUF1     130010   23335  23397  23433#
BUF2     130410   23338  23408  23434#
B1       050462   11546# 11557
B2       050464   11549#
B3       050504   11555  11557#
CARRY1   003146    1067   1069#
CC     = 177776     537#  6795*
CCBDON   103774   18636  18640#
CCB10    103772   18633  18635  18638#
```

```
CCB2    103746          18628#
CCCDON  070724          15610   15670#
CCC1    070176          15506#
CCC10   070462          15578#
CCC11   070506          15586#
CCC12   070532          15595#
CCC13   070556          15603#
CCC2    070222          15514#
CCC3    070246          15522#
CCC4    070272          15529#
CCC5    070316          15537#
CCC6    070342          15545#
CCC7    070366          15553#
CCC8    070412          15562#
CCC9    070436          15570#
CCDATO  062150          13864   13883   13903   13923   13942   13961   13972#
CCERR   020302           5710#
CCP0    062160          13859   13878   13895   13937   13956   13976#
CCP1    062170          13880   13980#
CCP10   062260          13925   14008#
CCP11   062270          13944   14012#
CCP12   062300          14016#
CCP2    062200          13861   13866   13984#
CCP3    062210          13915   13939   13988#
CCP4    062220          13958   13963   13992#
CCP5    062230          13919   13996#
CCP6    062240          13899   13905   14000#
CCP7    062250          13885   14004#
CCXDON  062310          13970   14021#
CCX12   061634          13888   13890#
CCX13   061644          13892   13895#
CCX14   061664          13900#
CCX15   061710          13907#  13910
CCX18   061716          13908   13910#
CCX19   061726          13912   13915#
CCX2    061524          13862#
CCX20   061746          13920#
CCX21   061772          13927#  13930
CCX24   062000          13928   13930#
CCX25   062010          13932   13935#
CCX26   062030          13940#
CCX27   062052          13946#  13949
CCX3    061546          13868#  13871
CCX30   062060          13947   13949#
CCX31   062070          13951   13954#
CCX32   062110          13959#
CCX33   062132          13965#  13968
CCX36   062140          13966   13968#
CCX6    061554          13869   13871#
CCX7    061564          13873   13876#
CCX8    061604          13881#
CCX9    061626          13887#  13890
CC1     001520            709*    711*  725#
CC2     020212           5684*   5692#  5695*  5701*
CC3     020176           5683*   5688#  5696*  5697    5699
CDONE   050736          11643   11645#
```

```
CHECK1  127622  23394#
CHECK2  127660  23405#
CHKAPT  050266   9515  11458#  21782
CIS     025212   6783   6837#
CISADR  024716   6766#  6775    6776
CISTRP  024702   6763#  6773
CLRCD   020174   5682#  5698    5702
CLR1    014032   4418   4419    4420   4422#
CMPARE  124036  22632  22635#
CMPREG  050104   9684   9691    9698   9705  11394#
CMPSUB  067612  15164  15174   15185  15196  15208  15219  15233  15247  15261  15274  15288  15302  15316
                15355#
CMPTMP  067706  15376  15387#
CMP1    014454   4609   4610    4612#
CMP2    014456   4611   4614#
CMP3    014500   4618   4619    4621#
CMP4    014502   4620   4623#
CMP5    014526   4627   4628    4629   4631#
CMP6    014530   4630   4633#
CMP7    014550   4636   4637    4639#
COMHLT  130006    676*  23431#
COM1    014572   4650   4651    4653#
CONCIS  024756   6764   6779#
CONT    001542    717    721     735#
CONT1   123172  22472#  22475
CONT42  126504  23183  23188#
CON1    020220   5693   5695#
CON2    020304   5709   5712#
CORH    023400   6470*  6483#
COUNT   025272   6868#  6869    6904*  6939*  6944  6968*  6973  6996*  7001  7024*  7029  7053*  7059
                 7083*  7088    7354*  7376*  7386  7413*  7422  7450*  7459
CPUHLT  002076    670*   809     816#
CTRAP   023356   6475#  6494
C15     050516  11572#
C2      050532  11577  11579#
C25     050540  11582#
C3      050556  11587  11589#
C35     050564  11592#
C4      050602  11597  11599#
C45     050606  11601#
C5      050624  11606  11608#
C55     050630  11611#
C6      050644  11616  11618#
C65     050652  11620#
C7      050670  11625  11627#
C75     050676  11629#
C8      050714  11634  11636#
C85     050720  11638#
DADTBL  126672  23244#  23272  23273  23279
DAERR   020056   5610   5612   5614   5616  5618  5620  5623#
DALTB1  044004  10405  10461#
DALTB2  044040  10419  10479#
DALTB3  044270  10499  10555#
DALTB4  044324  10513  10573#
DBE     024524   6712#  6716
DBE1    024532   6707   6716#
```

```
DBE2    024530              6709    6714#
DBE3    024544              6717    6719#
DBE4    024556              6720    6722#
DBE5    024566              6723    6725#
DDBBF0  104160             18695   18718#
DDBDON  104170             18704   18723#
DDBTP1  104140             18681   18709#
DDBTP2  104150             18685   18713#
DDB2    104074             18690#
DDB5    104136             18700   18703   18705#
DDCDON  110732             20019   20028#
DDCTP1  110716             20009   20015   20023#
DDC10   110730             20012   20016   20018   20026#
DDC2    110664             20011   20013#
DDDATO  063660             14212   14232   14252   14272   14291   14310   14330   14342#
DDDDON  071416             15762   15825#
DDDONE  064010             14340   14386#
DDD1    070730             15685#
DDD2    070770             15696#
DDD3    071030             15707#
DDD4    071070             15718#
DDD5    071130             15729#
DDD6    071170             15741#
DDD7    071230             15752#
DDONE   051104             11680   11703#
DDP0    063670             14234   14254   14346#
DDP1    063700             14207   14209   14229   14247   14350#
DDP2    063710             14227   14249   14354#
DDP3    063720             14267   14288   14358#
DDP4    063730             14305   14327   14362#
DDP5    063740             14307   14325   14366#
DDP6    063750             14269   14286   14370#
DDP7    063760             14274   14293   14374#
DDP8    063770             14312   14332   14378#
DDP9    064000             14214   14382#
DD10    063242             14236#  14239
DD11    063250             14237   14239#
DD12    063264             14242   14245#
DD13    063304             14250#
DD14    063326             14256#  14259
DD15    063334             14257   14259#
DD16    063350             14262   14265#
DD17    063370             14270#
DD18    063412             14276#  14279
DD2     063134             14210#
DD21    063420             14277   14279#
DD22    063430             14281   14284#
DD23    063450             14289#
DD24    063472             14295#  14298
DD27    063500             14296   14298#
DD3     063156             14216#  14219
DD30    063510             14300   14303#
DD31    063530             14308#
DD32    063552             14314#  14317
DD35    063560             14315   14317#
DD36    063574             14320   14323#
```

```
DD37    063614          14328#
DD38    063636          14334#  14337
DD41    063644          14335   14337#
DD6     063164          14217   14219#
DD7     063200          14222   14225#
DD8     063220          14230#
DEC1    013732           4368    4369   4370    4372#
DEC2    013734           4371    4374#
DEC3    013750           4377    4378   4380#
DEC4    013752           4379    4382#
DEC5    013766           4385    4386   4388#
DEC6    013770           4387    4390#
DEC7    014012           4394    4395   4396    4398#
DERR1   051040          11659   11684#
DERR2   051006          11658   11671#
DISPRE  000174            631#
DIVDSU  071274          15685   15696   15707   15718   15729   15741   15752   15791#
DIVDT   071406          15807   15811   15822#
DIVFSU  070606          15506   15514   15522   15529   15537   15545   15553   15562   15570   15578   15586   15595   15603
                        15638#
DIVFT   070714          15654   15667#
DL11W   023500           6505    6508#
DNMB0A  007352           2924    2925   2926    2928#
DNMB0B  007354           2927    2930#
DNMB2A  007506           2983    2984   2985    2987#
DNMB2B  007510           2986    2989#
DNMB2C  007516           2990    2992#
DNMB2D  007532           2994    2995   2997#
DNMB2E  007534           2996    2999#
DNMB2F  007544           3001    3003#
DNMB3A  007610           3020    3021   3022    3024#
DNMB3B  007612           3023    3026#
DNMB3C  007622           3027    3029#
DNMB3D  007640           3032    3033   3035#
DNMB3E  007642           3034    3037#
DNMB4A  007762           3075    3076   3077    3079#
DNMB4B  007764           3078    3081#
DNMB4C  007774           3082    3084#
DNMB4D  010004           3085    3087#
DNMB4E  010006           3086    3089#
DNMB4F  010014           3090    3092#
DNM03A  006752           2671    2672   2673    2675#
DNM03B  006754           2674    2677#
DNM03C  006764           2678    2681#
DNM1    006664           2638    2640#
DNM1A   007404           2942    2943   2944    2946#
DNM1B   007406           2945    2948#
DNM2    006672           2641    2643#
DNM2A   007436           2960    2961   2963#
DNM2B   007440           2962    2965#
DNM2C   007446           2968#
DNM2D   007450           2967    2970#
DNM3    006702           2645    2647#
DNM4    006716           2651    2653#
DNM4A   007706           3052    3053   3054    3056#
DNM4B   007710           3055    3058#
```

```
DNM4C   007716          3059    3061#
DNM5A   010056          3106    3107    3108    3110#
DNM5B   010060          3109    3112#
DNM5C   010070          3113    3115#
DNM6A   010132          3128    3129    3130    3132#
DNM6B   010134          3131    3134#
DNM6C   010144          3135    3137#
DNM7A   010210          3150    3151    3152    3154#
DNM7B   010212          3153    3156#
DNM7C   010222          3157    3159#
DOAGIN  131066         23448   23454#
DOPB2A  007172          2834    2836#
DOPB2B  007224          2857    2859#
DOP0A   006520          2570    2572#
DOP0B   006536          2577    2579#
DOP0C   006550          2582    2584#
DOP0D   006572          2591    2593#
DOP03A  006632          2614    2615    2616    2618#
DOP03B  006634          2617    2620#
DOP1    007072          2765    2767#
DOP2    007142          2809    2811#
DOP4    011204          3474#
DOP5    011252          3503#
DPAT3   055632         12873#
DTRAP1  023376          6481#   6493
DUMMY = 000000          6864#   7579    7599    7620    7641    7661    7682    7703
D1      050762         11661#  11676   11682
D2      050774         11664*  11665#
D3      050776         11666#  11684
D4      051002         11668#
D5      051010         11673#  11699
D6      051024         11674   11678#
D7      051034         11679   11681#
D8      051070         11693#
D9      051102         11698   11701#
EDONE   051152         11727   11730#
EEBBF0  104304         18764#
EEBBF1  104314         18736   18742   18747   18753   18768#
EEBDON  104326         18757   18774#
EEBTP1  104274         18735   18760#
EEB10   104324         18743   18750   18754   18756   18772#
EEB2    104234         18744#
EECDON  111040         20048   20061#
EECTP1  111014         20039   20040*  20054#  20056
EECTP2  111024         20038   20044   20056#
EEC10   111036         20041   20045   20047   20059#
EEC2    110762         20042#
EEDATO  064164         14404   14424   14436#
EEDONE  064246         14434   14461#
EEEDON  072150         15943   16003#
EEE1    071422         15840#
EEE10   071706         15912#
EEE11   071732         15920#
EEE12   071756         15928#
EEE13   072002         15936#
EEE2    071446         15848#
```

```
EEE3    071472          15856#
EEE4    071516          15864#
EEE5    071542          15872#
EEE6    071566          15880#
EEE7    071612          15888#
EEE8    071636          15896#
EEE9    071662          15904#
EEP0    064174          14406   14426   14440#
EEP1    064206          14399   14401   14445#
EEP2    064216          14449#
EEP3    064226          14419   14421   14453#
EEP4    064236          14457#
EERR0   051150          11714   11723   11728#
EE12    064150          14429   14431#
EE2     064034          14402#
EE3     064056          14408#  14411
EE6     064064          14409   14411#
EE7     064100          14414   14417#
EE8     064120          14422#
EE9     C64142          14428#  14431
EISEND  037220           9180    9182#
EMTA  = 104377            540#    869
ENDMSG  131247          23445   23500#
ENDPAS  131010          23257   23309   23421   23439#
ENT176  031102           7561    7568#
ENT51   027026           7160    7164#
ER      001536            718     720    732#
ERROR1  002062            733     761    763   765   767   769   775   777   779   790   792   794   796
                          798     804    806   808   814#  827   829   831   833   835   841   843   845
                          856     858    860   862   864   870   872   874   880   988  6214  6224  6235
                         6236    6246   6256  6266  6279
ERROR2  002346            875     880#   893   895   897   899   901   907   909   911   928   930   932
                          934     936    942   944   946   957   959   961   963   965   971   973   975
ERROR3  050312           9276   11471#
ERROR4  120162          11500   21839#  21847
ERROR5  122654          21892   22396#
ERROR6  124064          22405   22647#
ERROR7  126654          22666   23239#
ERROR8  127772          23259   23429#
ERRVEC= 000004           9247#  10224*  10237*  11511*  11658*  11754*  11787*  12668*  12968*  13030*  13096*  15014*  17885*
                        17947*  18008*  18061*  18108*  18154*  18743*  18794*  18841*  18892*  18943*  18995*  19041*  19090*
                        19134*  19940*  19974*  20012*  20041*  20074*  20106*  20139*  20173*  20967*  21006*  21045*  21083*
                        21124*  21164*  21204*  21848*
E1      051116          11716#
E3      051124          11718#
E4      051126          11719#
F     = 000063           6865#   7172    7189#  7206#  7223#  7240#  7257#  7274#  7291#  7308#  7325#  7342#
FDATI0  051426          11746   11768   11815#
FDATI1  051430          11816#
FDATI2  051432          11817#
FDATI3  051434          11818#
FDATI4  051436          11753   11762   11819#
FDATI5  051440          11820#
FDATI6  051442          11821#
FDATI7  051444          11822#
FDATO0  051450          11782   11801   11824#
```

```
FDAT01  051452          11825#
FDAT02  051454          11826#
FDAT03  051456          11827#
FDAT04  051460          11786   11795   11828#
FDAT05  051462          11829#
FDAT06  051464          11830#
FDAT07  051466          11831#
FDONE   051512          11811   11843#
FERR20  051424          11754   11787   11812#
FFBBF1  104442          18787   18793   18799   18805   18316#
FFBDON  104454          18809   18822#
FFBTP1  104432          18786   18812#
FFB10   104452          18794   18802   18806   18808   18820#
FFB2    104372          18796#
FFCDON  111140          20081   20093#
FFCTP1  111116          20072   20073*  20086#  20088
FFCTP2  111126          20071   20077   20088#
FFC10   111136          20074   20078   20080   20091#
FFC2    111072          20075#
FFDATO  064412          14479   14499   14512#
FFDONE  064472          14508   14538#
FFFDON  072502          16061   16124#
FFF1    072154          16018#
FFF2    072214          16029#
FFF3    072254          16040#
FFF4    072314          16051#
FFP0    064422          14494   14517#
FFP1    064432          14496   14521#
FFP2    064442          14474   14525#
FFP3    064452          14476   14529#
FFP4    064462          14481   14501   14533#
FF10    064402          14504   14506#
FF2     064272          14477#
FF3     064314          14483#  14486
FF4     064322          14484   14486#
FF5     064332          14488   14492#
FF6     064352          14497#
FF7     064374          14503#  14506
FINIE   127716          23416#
FINISH  025256           6759*   6789   6855#
FOVER   022600           6287    6299#
FPEXIT  120156          21783   21836#
FPHLT   120176            672*  21841#
FPP     025246           6759    6787   6851#
FPSTRT  050330           9275   11317  11496#
FPVECT= 000244          11484#  11509*  11659*  11714*  12663*  12693*  13273*  13351*  14735*  14757*  14793*  14815*  14849*
                        14871*  15411*  15459*  16225*  16370*  16500*  16651*  17688*  21847*

FXDAT0  051472          11747   11769   11802   11833#
FXDAT1  051474          11834#
FXDAT2  051476          11835#
FXDAT3  051500          11836#
FXDAT4  051502          11837#
FXDAT5  051504          11838#
FXDAT6  051506          11839#
FXDAT7  051510          11840#
F10     051340          11791#
```

```
F11     051342          11792#
F12     051356          11796   11798#
F13     051402          11804#  11807
F135    051366          11799   11801#
F14     051410          11805   11807#
F2      051200          11750#  11751
F22     051412          11808#
F3      051220          11758#
F4      051222          11759#
F5      051236          11763   11765#
F6      051304          11778   11780#
F7      051320          11783#  11784
GAND0   054100          12433#
GAND1   054102          12434#
GAND2   054104          12435#
GAND3   054106          12436#
GCMP    054030          12075   12101   12127   12153   12179   12205   12231   12257   12283   12309   12335   12361   12408#
GDAT00  054120          12067   12093   12119   12145   12171   12197   12223   12249   12275   12301   12327   12353   12395
                        12401   12408   12443#
GDAT01  054122          12444#
GDAT02  054124          12445#
GDAT03  054126          12446#
GDONE   054130          12378   12449#
GFLAG1  054054          12077   12079*  12103   12105*  12129   12131*  12155   12157*  12181   12183*  12207   12209*  12233
                        12235*  12259   12261*  12285   12287*  12311   12313*  12337   12339*  12363   12365*  12381   12420#
GFLAG2  054056          12421#
GGBBF0  104570          18834   18846   18852   18863#
GGBBF1  104600          18840   18867#
GGBDON  104612          18856   18873#
GGBTP1  104560          18833   18859#
GGB10   104610          18841   18849   18853   18855   18871#
GGB2    104520          18843#
GGCDON  111234          20115   20125#
GGCTP1  111220          20103   20104*  20111   20120#
GGC10   111232          20106   20110   20112   20114   20123#
GGC2    111166          20107#
GGDAT0  066130          14741   14771   14799   14827   14855   14885   14903#
GGDONE  066250          14901   14945#
GGER0   065376          14735   14763#  14793
GGER14  066034          14849   14877#
GGGDON  075266          16830   16915#
GGG1    074336          16706#
GGG10   074666          16787#
GGG11   074716          16796#
GGG12   074746          16805#
GGG13   074776          16814#
GGG14   075026          16823#
GGG2    074366          16715#
GGG3    074416          16724#
GGG4    074446          16733#
GGG5    074476          16742#
GGG6    074526          16751#
GGG7    074556          16760#
GGG8    074606          16769#
GGG9    074636          16778#
GGP1    066140          14908#
```

GGP2	066150	14794	14816	14850	14872	14912#								
GGP3	066160	14796	14818	14916#										
GGP4	066170	14920#												
GGP5	066200	14736	14738	14758	14760	14924#								
GGP6	066210	14743	14773	14801	14857	14928#								
GGP7	066220	14829	14933#											
GGP8	066230	14852	14874	14937#										
GGP9	066240	14887	14941#											
GG10	065472	14787	14791#											
GG11	065520	14797#												
GG12	065542	14803#	14806											
GG13	065550	14804	14806#											
GG14	065564	14809	14813#											
GG15	065612	14819#	14821											
GG16	065616	14815	14821#											
GG17	065652	14831#	14834											
GG18	065660	14832	14834#											
GG19	065710	14843	14847#											
GG2	065300	14739#												
GG20	065736	14853#												
GG21	065760	14859#	14862											
GG22	065766	14860	14862#											
GG23	066002	14865	14869#											
GG24	066030	14875#	14879											
GG25	066036	14871	14879#											
GG26	066072	14889#	14892											
GG27	066100	14890	14892#											
GG3	065322	14745#	14748											
GG4	065330	14746	14748#											
GG5	065344	14751	14755#											
GG6	065372	14761#	14765											
GG7	065400	14757	14765#											
GG8	065434	14775#	14778											
GG9	065442	14776	14778#											
GIN1	024770	6781#	6788	6812										
GIN2	025032	6792#	6813											
GIN3	025060	6790	6797#											
GOR0	054110	12438#												
GOR1	054112	12439#												
GOR2	054114	12440#												
GOR3	054116	12441#												
GPAT00	054060	12066	12118	12170	12222	12274	12326	12391	12423#					
GPAT01	054062	12424#												
GPAT02	054064	12425#												
GPAT03	054066	12426#												
GPAT10	054070	12092	12144	12196	12248	12300	12352	12386	12428#					
GPAT11	054072	12429#												
GPAT12	054074	12430#												
GPAT13	054076	12431#												
GRESET	054010	12087	12113	12139	12165	12191	12217	12243	12269	12295	12321	12347	12373	12401#
GSETUP	053732	12068	12094	12120	12146	12172	12198	12224	12250	12276	12302	12328	12354	12381#
GS1	053762	12391#	12405											
G1	052346	12070#	12089											
G10	052600	12130	12134#											
G11	052602	12133	12135#											
G12	052646	12148#	12167											

G13	052700	12156	12160#					
G14	052702	12159	12161#					
G15	052746	12174#	12193					
G16	053000	12182	12186#					
G17	053002	12185	12187#					
G2	052400	12078	12082#					
G20	053046	12200#	12219					
G21	053100	12208	12212#					
G22	053102	12211	12213#					
G23	053146	12226#	12245					
G24	053200	12234	12238#					
G25	053202	12237	12239#					
G26	053246	12252#	12271					
G27	053300	12260	12264#					
G3	052402	12081	12083#					
G30	053302	12263	12265#					
G31	053346	12278#	12297					
G32	053400	12286	12290#					
G33	053402	12289	12291#					
G34	053446	12304#	12323					
G35	053500	12312	12316#					
G36	053502	12315	12317#					
G37	053546	12330#	12349					
G4	052446	12096#	12115					
G40	053600	12338	12342#					
G41	053602	12341	12343#					
G42	053646	12356#	12375					
G43	053700	12364	12368#					
G44	053702	12367	12369#					
G5	052500	12104	12108#					
G6	052502	12107	12109#					
G7	052546	12122#	12141					
HADR	054622	12594*	12602	12611#				
HA1R	054676	12572	12596	12605	12625#			
HA1W	054626	12462	12472	12499	12595	12614#		
HA2R	054706	12576	12627#					
HA2W	054636	12476	12512	12616#				
HA3R	054716	12580	12629#					
HA3W	054646	12480	12525	12618#				
HA4R	054726	12584	12631#					
HA4W	054656	12484	12538	12620#				
HA5R	054736	12588	12633#					
HA5W	054666	12488	12551	12622#				
HCLR	054604	12468	12570	12605#				
HCLR1	054614	12607#	12608					
HCMP	054550	12494	12506	12519	12532	12545	12558	12594#
HCMP1	054570	12598#	12601					
HCMP2	054576	12599	12601#					
HDAT1	054746	12463	12636#					
HDAT2	054756	12638#						
HDAT3	054766	12640#						
HDAT4	054776	12642#						
HDAT5	055006	12644#						
HDONE	055016	12564	12647#					
HERE =	000000	5999#	6022*	6028*	6074			
HFLAG	054624	12461*	12562	12566*	12612#			

```
HHBBF0  104736          18885   18897   18912   18914#
HHBBF1  104746          18891   18902   18918#
HHBDON  104760          18906   18924#
HHBTP1  104716          18884   18909#
HHB10   104756          18892   18900   18903   18905   18922#
HHB2    104656          18894#
HHCDON  111344          20148   20159#
HHCTP1  111322          20136   20137*  20153#
HHCTP2  111332          20135   20144   20155#
HHC10   111342          20139   20143   20145   20147   20157#
HHC2    111270          20140#
HHDATO  065076          14568   14592   14615   14635   14655   14667#
HHDONE  065246          14665   14719#
HHHDON  076502          17098   17181#
HHH1    075272          16931#
HHH10   076042          17048#
HHH11   076112          17061#
HHH12   076162          17074#
HHH13   076232          17087#
HHH2    075342          16944#
HHH3    075412          16957#
HHH4    075462          16970#
HHH5    075532          16983#
HHH6    075602          16996#
HHH7    075652          17009#
HHH8    075722          17022#
HHH9    075772          17035#
HHP0    065106          14563   14671#
HHP1    065116          14565   14675#
HHP10   065226          14637   14711#
HHP11   065236          14657   14715#
HHP2    065126          14570   14679#
HHP3    065136          14683#
HHP4    065146          14594   14687#
HHP5    065156          14587   14650   14652   14691#
HHP6    065166          14589   14695#
HHP7    065176          14617   14699#
HHP8    065206          14610   14630   14632   14703#
HHP9    065216          14612   14707#
HH10    064626          14597   14599#
HH11    064642          14602   14607#
HH12    064662          14613#
HH13    064704          14619#  14622
HH16    064712          14620   14622#
HH17    064726          14625   14628#
HH18    064746          14633#
HH19    064770          14639#  14642
HH2     064516          14566#
HH20    064776          14640   14642#
HH21    065012          14645   14648#
HH22    065032          14653#
HH23    065054          14659#  14662
HH24    065062          14660   14662#
HH3     064540          14572#  14575
HH6     064546          14573   14575#
HH7     064556          14577   14584#
```

```
HH8      064576            14590#
HH9      064620            14596#  14599
HICORE   023272             6454#   6472
HLT    = 000000              518#
HSTD     054472            12492   12505   12518   12531   12544   12557   12570#
HXDATO   067002            14969   14997   15031   15048   15072   15094   15108#
HXDONE   067112            15105   15148#
HXER9    066506            15014   15022#
HXP1     067012            14960   14985   15063   15067   15075   15091   15096   15113#
HXP2     067022            14962   14964   14988   14991   15028   15118#
HXP3     067032            15122#
HXP4     067042            15045   15126#
HXP5     067052            15051   15130#
HXP6     067062            15026   15043   15089   15134#
HXP7     067072            14971   15033   15139#
HXP8     067102            15000   15143#
HX10     066404            14992   14995#
HX11     066426            15002#  15005
HX14     066434            15003   15005#
HX15     066450            15009   15012#
HX16     066466            15016#
HX165    066472            15017#
HX17     066510            15021   15024#
HX18     066530            15029#
HX19     066550            15035#  15038
HX2      066274            14963#
HX20     066556            15036   15038#
HX22     066600            15046#
HX23     066620            15053#  15056
HX26     066626            15054   15056#
HX28     066652            15068#
HX29     066674            15077#  15080
HX3      066306            14965   14967#
HX30     066702            15078   15080#
HX31     066716            15084   15087#
HX32     066736            15092#
HX33     066760            15098#  15101
HX34     066766            15099   15101#
HX4      066326            14973#  14976
HX7      066334            14974   14976#
HX8      066350            14979   14982#
HX9      066372            14990#
H1       054134            12461#
H10      054376            12541#  12546
H11      054430            12554#  12559
H12      054462            12563   12566#
H2       054154            12465#  12466
H3       054164            12470#  12567
H4       054236            12492#
H5       054260            12502#  12507
H6       054312            12515#  12520
H7       054344            12528#  12533
IDATI0   052004            11857   11863   11878   11888   11893   11919   11942#
IDATI1   052006            11943#
IDATI2   052010            11916*  11944#
IDATI3   052012            11917*  11945#
```

```
IDAT00  051774          11874   11877   11913   11918   11937#
IDAT01  051776          11938#
IDAT02  052000          11939#
IDAT03  052002          11940#
IDONE   052014          11925   11947#
IIBBF0  105104          18936   18948   18963   18965#
IIBBF1  105114          18942   18953   18969#
IIBDON  105126          18957   18976#
IIBTP1  105064          18935   18960#
IIB10   105124          18943   18951   18954   18956   18974#
IIB2    105024          18945#
IICDON  111410          20185   20189#
IIC2    111366          20178#
IIC20   111406          20173   20186#
IIIDON  073004          16175   16256#
III1    072506          16139#
III2    072536          16149#
III3    072566          16158#
III4    072616          16167#
IJMP    C12754           4066    4069#
IJMP4   012572           4027    4030#   4038
IJMP5   012724           4058    4061#
ILLA  = 004700            535#    6265
ILLB  = 000100            536#    6278
INC1    013646           4336    4337    4339#
INC2    013650           4338    4341#
INC3    013672           4345    4346    4347    4349#
INC4    013674           4348    4352#
INC5    013710           4355    4356    4358#
INST    025260           6234    6791*   6792*   6796    6804    6811    6856#
IOHAND  127546          23347   23354   23363   23371   23381#
IPAT10  051754          11858   11887   11927#
IPAT11  051756          11928#
IPAT12  051760          11929#
IPAT13  051762          11930#
IPAT20  051764          11866   11896   11932#
IPAT21  051766          11933#
IPAT22  051770          11934#
IPAT23  051772          11935#
ITRAP5= 000004            520#
I1      051516          11855#
I105    051616          11881   11883#
I12     051620          11887#
I13     051634          11890#  11891
I14     051662          11902#
I15     051664          11903#
I16     051666          11904#
I17     051702          11908   11910#
I2      051534          11860#  11861
I20     051742          11921#  11924
I23     051750          11922   11924#
I3      051560          11870#
I4      051562          11871#
I5      051564          11872#
I6      051610          11880#  11883
JBUF0   055430          12776   12792#
```

```
JBUF1   055432          12793#
JBUF2   055434          12794#
JBUF3   055436          12795#
JDATI0  055440          12766   12779   12787   12797#
JDATI1  055442          12798#
JDATI2  055444          12799#
JDATI3  055446          12800#
JDATO0  055450          12773   12780   12802#
JDATO   055460          12763   12807#
JDATO1  055452          12803#
JDATO2  055454          12804#
JDATO3  055456          12805#
JDATI1  055462          12808#
JDATI2  055464          12809#
JDATI3  055466          12810#
JDONE   055470          12790   12813#
JJBBFO  105246          18988   18994   19000   19005   19016#
JJBDON  105260          19009   19022#
JJBTP1  105234          18987   19012#
JJB10   105256          18995   19003   19006   19008   19020#
JJB2    105172          18997#
JJJDON  073366          16322   16401#
JJJ1    073010          16272#
JJJ2    073054          16285#
JJJ3    073120          16298#
JJJ4    073164          16311#
JMPCK   012756           4069    4071#
JMPERR  020142           5659    5660    5661    5663#
JMPSEQ  012770           4018*   4024    4028*   4032    4035*   4041    4045*   4048    4052*   4055    4059*   4063    4067*
                         4071    4074#
JMPT    020132           5658    5659#
JMP2    012574           4019    4032#
JMP2A   012604           4033    4035#
JMP3    012536           4021#   4036
JMP3A   012546           4022    4024#
JMP3B   012560           4025    4027#
JMP4    012616           4030    4038#
JMP4A   012626           4039    4041#
JMP4B   012640           4042    4044#
JMP5    012700           4044    4055#
JMP5A   012712           4056    4058#
JMP6    012652           4048#   4061
JMP6A   012664           4049    4051#
JMP7    012726           4051    4063#
JMP7A   012740           4064    4066#
JSRCK   013340           4183    4186#
JSRCKA  013334           4180    4183#
JSRCK1  013356           4097    4187    4190#
JSRSEQ  013336           4101*   4110    4122*   4125    4137*   4141    4147*   4151    4159*   4161    4169*   4173    4179*
                         4184#   4186
JSR0    012774           4097#
JSR1    013000           4096    4099#
JSR1A   013022           4107#   4127
JSR2    013076           4100    4125#   4143
JSR2A   013126           4126    4128    4130    4133#
JSR2B   013130           4132    4135#
```

```
JSR3    013024          4110#   4118    4138
JSR3A   013064          4111    4113    4115    4117    4120#
JSR3B   013066          4119    4122#
JSR4    013152          4112    4123    4141#
JSR4A   013170          4142    4145#
JSR4B   013172          4144    4147#
JSR5    013242          4148    4161#   4165    4175
JSR5A   013266          4162    4164    4167#
JSR5B   013270          4166    4169#
JSR6    013204          4151#   4163    4182
JSR6A   013230          4152    4154    4157#
JSR6AD  013332          4155    4170    4182#   4188
JSR6B   013232          4156    4159#
JSR7    013302          4153    4160    4173#
JSR7A   013320          4174    4177#
JSR7B   013322          4176    4179#
J1      055330         12760#
J2      055346         12769#
J3      055350         12770#
J4      055352         12771#
J5      055406         12782#  12785
J6      055414         12783   12785#
J7      055426         12788   12790#
KBUF0   055604         12830   12839   12850   12860#
KBUF1   055606         12861#
KBUF2   055610         12862#
KBUF3   055612         12863#
KDATI0  055574         12842   12855#
KDATI1  055576         12856#
KDATI2  055600         12857#
KDATI3  055602         12858#
KDATO0  055614         12836   12843   12865#
KDATO1  055616         12866#
KDATO2  055620         12867#
KDATO3  055622         12868#
KDONE   055634         12853   12875#
KERSTK= 001000           538#   9352    9356    9682    9689    9696    9703   10226   10409   10430   10503   10524   10606
                       10653   10658   10711   10728   10771   10817   10899   11166   11267   11296   11315
KFIRST  124060         22633*  22635   22643#
KIPAR0= 172340          9232#   9394    9534    9606    9688    9733    9797   10865*  11371   11408
KIPAR1= 172342          9233#  10866*
KIPAR2= 172344          9234#  10867*
KIPAR3= 172346          9235#  10112   10258*  10594*  10868*
KIPAR4= 172350          9236#   9814*   9838*   9862*   9887*   9909*   9938*   9962*   9986*  10010*  10039*  10085*  10092*
                       10259*  10595*  10869*  10888*  11004*  11152*
KIPAR5= 172352          9237#  10040*  10086*  10093*  10631*
KIPAR6= 172354          9238#  10236*
KIPAR7= 172356          9239#  10870*
KIPDR0= 172300          9221#   9406    9568    9643    9680    9722    9804   10116   10119   10132   10874   11367   11395
KIPDR1= 172302          9222#
KIPDR2= 172304          9223#
KIPDR3= 172306          9224#  10262*  10403*  10596*
KIPDR4= 172310          9225#  10083*  10090*  10269*  10328*  10412*  10423*  10506*  10517*  10597*  10622*  10632*  10699*
                       10887*  10893*  11002*  11027*  11029*  11043*  11045*  11057*  11059*  11074*  11076*  11091*  11093*
                       11107*  11109*  11125*  11127*  11151*
KIPDR5= 172312          9226#  10084*  10091*  10404*  10633*  10700*
```

```
KIPDR6= 172314          9227#  10223*  10227   10235*
KIPDR7= 172316          9228#  10213*  10216   10762*  10791*
KKBBF0  105406         19034   19046   19061   19063#
KKBBF1  105416         19040   19051   19067#
KKBDON  105430         19055   19073#
KKBTP1  105366         19033   19058#
KKB10   105426         19041   19049   19052   19054   19071#
KKB2    105324         19043#
KKCDON  112506         20408   20472#
KKC1    111512         20244#
KKC10   111744         20305#
KKC11   111772         20313#
KKC12   112020         20321#
KKC13   112046         20329#
KKC14   112074         20337#
KKC15   112122         20345#
KKC16   112150         20353#
KKC17   112176         20361#
KKC2    111540         20254#
KKC20   112224         20369#
KKC21   112252         20377#
KKC22   112300         20385#
KKC23   112326         20393#
KKC24   112354         20401#
KKC3    111566         20264#
KKC4    111614         20273#
KKC5    111642         20281#
KKC6    111670         20289#
KKC7    111716         20297#
KKKDON  073720         16455   16539#
KKK1    073372         16417#
KKK3    073422         16427#
KKK4    073452         16437#
KKK5    073502         16447#
KPAT0   055624         12827   12870#
KPAT1   055626         12871#
KPAT2   055630         12872#
KWSTRT  122672         21890   21930   22394   22401#
KX2     055512         12832#
KX3     055514         12833#
KX4     055516         12834#
KX5     055552         12845#  12848
KX6     055560         12846   12848#
KX7     055572         12851   12853#
K1      021240          6001#   6083*   6084    6087    6092*   6093    6096    6100*   6102    6109*   6111    6118*   6120
                        6123
K10     021256          6008#   6138    6142    6145    6149    6153    6157
K11     021260          6009#
K12     021262          6010#
K2      021242          6002#
K3      021244          6003#
K4      021246          6004#
K5      021250          6005#   6082*   6085    6091*   6101*   6103    6105    6110*   6112    6114    6119*   6121
K6      021252          6006#   6094    6142
K7      021254          6007#   6130    6134    6138    6145    6157
LDATI0  056010         12893   12900   12918   12938#
```

```
LDATI1  056012          12939#
LDATI2  056014          12913   12916*  12940#
LDATI3  056016          12917*  12941#
LDATO0  056022          12910   12919   12943#
LDATO1  056024          12944#
LDATO2  056026          12945#
LDATO3  056030          12946#
LDCDSU  113170          20485   20495   20505   20515   20525   20535   20545   20555   20566   20576   20618#
LDCFSU  112404          20244   20254   20264   20273   20281   20289   20297   20305   20313   20321   20329   20337   20345
                        20353   20361   20369   20377   20385   20393   20401   20441#
LDCT    112476          20449   20454   20469#  20625   20631
LDONE   056032          12926   12948#
LDXSUB  114430          20659   20672   20685   20698   20712   20725   20739   20752   20765   20778   20791   20804   20818
                        20832   20846   20860   20907#
LDXT    114556          20923   20925   20944#
LKS     120256          21884#  22414   22427   22431*  22432   22435*  22436   22444*  22446   22461   22464*  22465   22467*
                        22468   22472   22489*  22490*  22491   22497*  22500   22507*  22508*  22509   22516*  22538*  22539*
                        22540   22553*  22570*  22571*  22572   22577*  22591*  22592*  22593   22595*  22598*  22616*  22621*
                        22622   22624*  22625   22628*  23339*  23386*
LKSTST  122722          22407#
LLBBFO  105534          19084   19092*  19095   19109#
LLBDON  105546          19102   19116#
LLBTP1  105524          19083   19105#
LLB10   105544          19090   19098   19101   19114#
LLB2    105470          19092#
LLCDON  113262          20585   20645#
LLC1    112512          20485#
LLC10   113034          20555#
LLC11   113072          20566#
LLC12   113130          20576#
LLC2    112550          20495#
LLC3    112606          20505#
LLC4    112644          20515#
LLC5    112702          20525#
LLC6    112740          20535#
LLC7    112776          20545#
LLLDON  074332          16605   16689#
LLL1    073724          16555#
LLL2    073770          16568#
LLL3    074034          16581#
LLL4    074100          16594#
LPAT10  055766          12890   12928#
LPAT11  055770          12929#
LPAT12  055772          12930#
LPAT13  055774          12931#
LPAT20  055776          12894   12933#
LPAT21  056000          12934#
LPAT22  056002          12935#
LPAT23  056004          12936#
LTCHLT  124100            673*  22649#
L1      055640          12887#
L2      055700          12904#
L3      055702          12905#
L4      055704          12907#
L5      055754          12922#  12925
L6      055762          12923   12925#
```

```
MBDM2A  010344          3228    3229    3231#
MBDM2B  010346          3230    3233#
MBDM2C  010356          3234    3236#
MBDM2D  010370          3237    3238    3240#
MBDM2E  010372          3239    3242#
MBDM2F  010402          3243    3245#
MBDM4A  010612          3326    3328#
MBDM4B  010622          3329    3331#
MBDM4C  010634          3332    3333    3335#
MBDM4D  010636          3334    3337#
MBDM4E  010644          3338    3340#
MDAT00  056160         12980   12983   13004#
MDAT01  056162         13005#
MDAT02  056164         13006#
MDAT03  056166         13007#
MDM1A   010254          3178    3179    3181#
MDM1B   010256          3180    3183#
MDM2A   010304          3202    3203    3205#
MDM2B   010306          3204    3207#
MDM2C   C10314          3210#
MDM2D   010316          3209    3212#
MDM3A   010440          3262    3263    3265#
MDM3B   010442          3264    3267#
MDM3C   010452          3268    3270#
MDM3D   010464          3271    3273#
MDM3E   010504          3275    3277#
MDM4A   010546          3297    3298    3300#
MDM4B   010550          3299    3302#
MDM4C   010556          3303    3305#
MDM5A   010676          3361    3362    3364#
MDM5B   010700          3363    3366#
MDM5C   010710          3367    3369#
MDM5D   010720          3370    3372#
MDM5E   010740          3375    3377#
MDM6A   010772          3397    3398    3400#
MDM6B   010774          3399    3402#
MDM6C   011004          3403    3405#
MDM6D   011016          3406    3408#
MDM6E   011040          3411    3413#
MDM7A   011076          3432    3433    3435#
MDM7B   011100          3434    3437#
MDM7C   011110          3438    3440#
MDM7D   011122          3441    3443#
MDM7E   011140          3445    3447#
MDONE   056170         12992   13009#
MERR3   056076         12968   12978#
MFPIO   017630          5554    5557#
MFPIOA  017650          5559    5561#
MFPS1   016720          5328    5331#
MFPS2A  016772          5347    5348    5349    5351#
MFPS2B  016774          5350    5353#
MFPS2C  017006          5354    5356#
MFPS3A  017046          5368    5369    5370    5372#
MFPS3B  017050          5371    5374#
MFPS3C  017062          5375    5377#
MFPS4A  017122          5389    5390    5391    5393#
```

```
MFPS4B  017124          5392    5395#
MFPS4C  017136          5396    5398#
MFPS5A  017176          5410    5411    5412    5414#
MFPS5B  017200          5413    5416#
MFPS5C  017212          5417    5419#
MFPS6A  017254          5431    5432    5433    5435#
MFPS6B  017256          5434    5437#
MFPS6C  017270          5438    5440#
MFPS7A  017332          5452    5453    5454    5456#
MFPS7B  017334          5455    5458#
MFPS7C  017346          5459    5461#
MFPT  = 000007          5885#   5887
MMBBF0  105662         19128   19139   19152   19154#
MMBBF1  105672         19136*  19158#
MMBDON  105704         19146   19165#
MMBTP1  105642         19127   19149#
MMB10   105702         19134   19142   19145   19163#
MMB2    105606         19136#
MMCDON  114566         20872   20947#
MMC1    113266         20659#
MMC10   113700         20752#
MMC11   113746         20765#
MMC12   114014         20778#
MMC13   114062         20791#
MMC14   114130         20804#
MMC15   114176         20818#
MMC16   114244         20832#
MMC17   114312         20846#
MMC2    113334         20672#
MMC20   114360         20860#
MMC3    113402         20685#
MMC4    113450         20698#
MMC5    113516         20712#
MMC6    113564         20725#
MMC7    113632         20739#
MMMDON  077142         17251   17341#
MMM1    076506         17197#
MMM2    076542         17208#
MMM3    076576         17219#
MMM4    076632         17230#
MMM5    076666         17241#
MMUHLT  050326           671*  11473#
MMUTST  037252          9182    9268#
MMVEC = 000250          9188#   9269*   9270*  10053*  10096*  10268*  10314*  10327*  10366*  10408*  10421*  10456*  10502*
                       10515*  10550*  10600*  10623*  10650*  10656*  10701*  10892*  10995*  11006*  11136*  11159*  11257*
MODDD0  077626         17467   17471   17499#
MODDD1  077636         17469   17479   17502#
MODDOV  077432         17357   17372   17386   17401   17449#
MODDSU  076306         16931   16944   16957   16970   16983   16996   17009   17022   17035   17048   17061   17074   17087
                       17133#
MODDT0  076462         17150   17154   17175#
MODDT1  076472         17152   17162   17178#
MODFD0  077122         17305   17309   17335#
MODFD1  077132         17307   17316   17338#
MODFOV  076726         17197   17209   17219   17230   17241   17287#
MODFSU  075062         16706   16715   16724   16733   16742   16751   16760   16769   16778   16787   16796   16805   16814
```

```
                    16823   16866#
MODFT0  075236      16883   16887   16906#
MODFT1  075246      16885   16894   16909#
MODP1   075256      16871   16912#  17138   17292   17454
MOR0    020336       5716    5727#
MOR1    020360       5737    5740#
MOR2    020414       5755    5758#
MOR3    020462       5775    5778#
MOR4    020474       5779    5782#
MOR5    020506       5784    5787#
MOR6    020552       5802    5805#
MOR7    020564       5807    5811#
MOR8    020576       5812    5815#
MOV1    013426       4236    4237    4239#
MOV2    013430       4238    4242#
MOV3    013446       4245    4246    4248#
MPAT10  056140      12964   12994#
MPAT11  056142      12995#
MPAT12  056144      12996#
MPAT13  056146      12997#
MPAT20  056150      12999#
MPAT21  056152      13000#
MPAT22  056154      13001#
MPAT23  056156      13002#
MRK1    016234       5164    5169#
MRK2    016256       5169    5170    5171    5173    5176#
MRK3    016260       5175    5178#
MRK4    016302       5181    5183#
MRK5    016306       5182    5185#
MRK6    016322       5186    5189#
MTFLAG  000172        630#    661*   23475
MTPI0   017716       5574    5577#
MTPS1   016346       5209    5211#
MTPS1A  016366       5215    5216    5217    5219#
MTPS2   016416       5231    5233#
MTPS3   016462       5247    5249#
MTPS4   016524       5262    5264#
MTPS5   016560       5277    5279#
MTPS6   016624       5292    5294#
MTPS7   016670       5307    5309#
MULDSU  072360      16018   16029   16040   16051   16088#
MULDT   072472      16106   16110   16122#
MULFSU  072032      15840   15848   15856   15864   15872   15880   15888   15896   15904   15912   15920   15928   15936
                    15971#
MULFT   072140      15987   16000#
M1      056036      12961#
M15     056056      12970#
M2      056062      12973#
M3      056064      12974#
M4      056066      12975#
M5      056122      12985   12987#
M6      056126      12988#  12991
M7      056134      12989   12991#
M8      056100      12977   12980#
N     = 000307       6866#   7574    7594#   7614#   7636#   7656#   7676#   7698#   7718#   7741#   7746    7769#   7792#
                     7815#   7838#   7861#   7884#   7907#   7930#   7953#   7976#   7990    8011#   8032#   8053#   8076#
```

```
                    8097#  8118#  8141#  8162#  8183#  8204#  8227#  8248#  8269#  8290#  8313#  8334#  8355#
                    8378#  8382   8403#  8424#  8445#  8466#  8487#  8508#  8529#  8550#  8571#  8592#  8598
                    8619#  8640#  8663#  8684#  8705#  8728#  8749#  8770#  8791#  8814#  8835#  8856#  8879#
                    8900#  8921#  8939#  8955#  8959   8980#  9001#  9022#  9043#  9064#  9085#  9106#  9127#
                    9148#  9169#
NATBF1  110462      19878  19884  19901  19918#
NATINS  110366      19888  19891#
NATSUB  110312      19667  19686  19705  19724  19743  19762  19781  19800  19819  19875#
NBR     001530        717*   720*   729#
NDATI0  056334      13044  13060  13070#
NDATI1  056336      13071#
NDATI2  056340      13072#
NDATI3  056342      13073#
NDATO0  056272      13037  13043  13053#
NDATO1  056274      13054#
NDATO2  056276      13055#
NDATO3  056300      13056#
NDONE   056344      13051  13075#
NEGAT   026426       6955   6983   7011   7039   7069   7098   7102#
NEG00   004530       1697   1698   1699   1701#
NEG01   004532       1700   1704#
NEG02   004540       1705   1708#
NEG03   004554       1710   1711   1712   1714#
NEG04   004556       1713   1716#
NEG1    014362       4574   4575   4576   4578#
NEG10   004604       1727   1728   1729   1731#
NEG11   004606       1730   1734#
NEG12   004616       1735   1737#
NEG13   004632       1739   1740   1741   1743#
NEG14   004634       1742   1745#
NEG2    014364       4577   4580#
NEG20   004664       1756   1757   1758   1760#
NEG21   004666       1759   1762#
NEG22   004706       1768   1770#
NEG3    014406       4584   4585   4586   4588#
NEG30   005116       1894   1895   1896   1898#
NEG31   005120       1897   1900#
NEG32   005126       1901   1903#
NEG33   005144       1906   1908#
NEG34   005152       1909   1911#
NEG4    014410       4587   4590#
NEG40   005406       2045   2046   2047   2049#
NEG41   005410       2048   2051#
NEG42   005416       2052   2054#
NEG5    014430       4594   4595   4596   4598#
NEG50   005454       2069   2070   2071   2073#
NEG51   005456       2072   2075#
NEG52   005464       2076   2078#
NEG60   005524       2093   2094   2095   2097#
NEG61   005526       2096   2099#
NEG70   005560       2111   2112   2113   2115#
NEG71   005562       2114   2117#
NERR0   056264      13030  13048#
NNBBF0  106016      19185  19187  19206#
NNBDON  106030      19195  19213#
NNBTP1  105776      19178  19198#
```

```
NNBTP2  106006          19188   19202#
NNB10   106026          19191   19194   19211#
NNB2    105724          19180#
NNCDON  114700          20978   20987#
NNCTB0  114662          20962   20981#
NNCTB1  114666          20969   20972   20974   20976   20982#
NNC10   114676          20967   20973   20975   20977   20984#
NNC2    114630          20971#
NNNDON  077646          17414   17505#
NNN1    077146          17357#
NNN2    077222          17372#
NNN3    077276          17386#
NNN4    077352          17401#
NODL    023466           6503    6506#
NODL2   024114           6589#
NOP   = 000240            504     531#
NOR     023266           6452#   6455
NOSUB   023362           6474    6477#
NPAT10  056324          13025   13065#
NPAT11  056326          13066#
NPAT12  056330          13067#
NPAT13  056332          13068#
NPAT20  056312          13028   13040   13060#
NPAT21  056314          13061#
NPAT22  056316          13062#
NPAT23  056320          13063#
N1      056174          13022#
N12     056244          13041   13043#
N13     056260          13046#  13050
N14     056266          13047   13050#
N2      056220          13033#
N3      056222          13034#
N4      056224          13035#
ODATI0  056510          13112   13126   13137#
ODATI1  056512          13138#
ODATI2  056514          13139#
ODATI3  056516          13140#
ODATO0  056446          13104   13111   13120#
ODATO1  056450          13121#
ODATO2  056452          13122#
ODATO3  056454          13123#
ODONE   056520          13118   13142#
OERR0   056416          13096   13109#
OOBDON  106160          19246   19260#
OOBTP1  106136          19224   19229   19236   19242   19249#
OOBTP2  106146          19225   19237   19253#
OOB10   106156          19240   19243   19245   19258#
OOB2    106072          19234#
OOCDON  115012          21017   21026#
OOCTB0  114774          21001   21020#
OOCTB1  115000          21008   21011   21013   21015   21021#
OOC10   115010          21006   21012   21014   21016   21023#
OOC2    114742          21010#
OODON   100640          17714   17717#
OOOT    100574          17688   17699#
OOO2    100572          17695#  17700
```

```
0003    100606          17704#
0004    100632          17711   17713#
OPAT10  056500          13091   13132#
OPAT11  056502          13133#
OPAT12  056504          13134#
OPAT13  056506          13135#
OPAT20  056464          13107   13126#
OPAT21  056466          13094   13127#
OPAT22  056470          13128#
OPAT23  056472          13129#
OPAT24  056474          13130#
OVDNTT  073356          16381   16383   16398#
OVDTT   074322          16668   16670   16686#
OVFNTT  072774          16236   16238   16253#
OVFTT   073710          16517   16519   16536#
OVUNDN  073234          16272   16285   16298   16311   16362#
OVUNDT  074150          16555   16568   16581   16594   16643#
OVUNFN  072652          16139   16149   16158   16167   16217#
OVUNFT  073536          16417   16427   16437   16447   16493#
01      056350          13088#
012     056420          13108   13111#
013     056434          13114#  13117
014     056442          13115   13117#
02      056374          13100#
03      056376          13101#
04      056400          13102#
PCN01   020612           5831#   5832
PCN1    024370           6656#   6658
PCN2    020630           5841#   5842
PCN3    020660           5852#   5853
PCN4    020704           5862#   5863
PCN5    020726           5872#   5873
PDATI0  056622          13161   13169   13175   13184#
PDATI1  056624          13185#
PDATI2  056626          13186#
PDATI3  056630          13187#
PDATO0  056632          13166   13170   13189#
PDATO1  056634          13190#
PDATO2  056636          13191#
PDATO3  056640          13192#
PDONE   056642          13178   13194#
PDRTB1  044022          10406   10470#
PDRTB2  044054          10420   10487#
PDRTB3  044306          10500   10564#
PDRTB4  044340          10514   10581#
PPAT10  056612          13158   13179#
PPAT11  056614          13180#
PPAT12  056616          13181#
PPAT13  056620          13182#
PPBDON  106310          19294   19308#
PPBTP1  106266          19271   19276   19284   19290   19297#
PPBTP2  106276          19272   19285   19301#
PPB10   106306          19288   19291   19293   19306#
PPB2    106222          19281#
PPCDON  115124          21056   21064#
PPCTB0  115106          21040   21059#
```

```
PPCTB1  115112          21047   21050   21052   21054   21060#
PPC10   115122          21045   21051   21053   21055   21062#
PPC2    115054          21049#
PPPBF0  100760          17734   17766#
PPPBF1  100774          17744   17750   17754   17769#
PPPDON  101022          17764   17777#
PPPTP1  101010          17741   17755   17772#
PPP10   101020          17757   17759   17761   17763   17775#
PPP2    100660          17736#  17737
PPP3    100710          17748#
PPP4    100722          17751   17754#
PS    = 177776            507#  1376*   1377    1385*   1386    1394*   1395    1403*   1404    5191#   5208    5228*   5230
                         5244*   5246    5259*   5261    5276    5289*   5291    5304*   5306    5325*   5333*   5345*   5366*
                         5387*   5408*   5429*   5450*   5482*   5484    5498*   5504*   5506*   5520*   5524*   5526*   5531*
                         5533*   5537*   5550*   5553    5555*   5557*   5567*   5569*   5573    5575*   5577*   5637*   5638*
                         5639    5642*   5644*   5689    5705    5914*   5917*   5926*   5932*   9243    21791*  23270*  23311*
                        23342*
PSW   = 177776           9243#   9305*   9306*   9307    9317*   9323*   9325*   9326    9335*   9337*   9338    9351*   9353*
                         9355*   9488*   9498*   9883    9905    10036*  10060   10077   10087*  10089*  10159*  10204*  10272
                        10305*  10311*  10313*  10331   10363*  10365*  10710*  10715*  10727*  10730*  10732*  10742*  10746
                        10748*  10807*  10816*  10818*  10820*  10896*  10912*  10923*  10934*  10944*  10957*  10970*  10982*
                        11009*  11020*  11039*  11054*  11070*  11086*  11103*  11119*  11150*  11161*  11175*  11187*  11198*
                        11208*  11220*  11233*  11245*  11258*  11264*  11295*  11329   11331   11552
PSWORD  025274           6870#   6871    6935*   6936    6964*   6965    6992*   6993    7020*   7021    7049*   7050    7079*
                         7080    7178*   7179    7195*   7196    7212*   7213    7229*   7230    7246*   7247    7263*   7264
                         7280*   7281    7297*   7298    7314*   7315    7331*   7332    7372*   7373    7409*   7410    7446*
                         7447    7583*   7584    7603*   7604    7624*   7625    7645*   7646    7665*   7666    7686*   7687
                         7707*   7708    7727*   7728    7755*   7756    7778*   7779    7801*   7802    7824*   7825    7847*
                         7848    7870*   7871    7893*   7894    7916*   7917    7939*   7940    7962*   7963    7997*   7998
                         8018*   8019    8039*   8040    8061*   8062    8083*   8084    8104*   8105    8126*   8127    8148*
                         8149    8169*   8170    8190*   8191    8212*   8213    8234*   8235    8255*   8256    8276*   8277
                         8298*   8299    8320*   8321    8341*   8342    8363*   8364    8389*   8390    8410*   8411    8431*
                         8432    8452*   8453    8473*   8474    8494*   8495    8515*   8516    8536*   8537    8557*   8558
                         8578*   8579    8606*   8607    8627*   8628    8649*   8650    8671*   8672    8692*   8693    8714*
                         8715    8736*   8737    8757*   8758    8778*   8779    8800*   8801    8822*   8823    8843*   8844
                         8865*   8866    8887*   8888    8908*   8909    8930*   8931*   8932    8947*   8948*   8949    8967*
                         8968    8988*   8989    9009*   9010    9030*   9031    9051*   9052    9072*   9073    9093*   9094
                         9114*   9115    9135*   9136    9156*   9157
PUSRM = 030000            511#
PWRDN   131130            656   23470#  23473
PWRMSG  131230          23477   23497#
PWRUP   131140          23470   23473#
P1      056524          13155#
P2      056540          13163#  13164
P3    = 056542          13164#
P4      056544          13166#
P5      056566          13171#  13174
QDATI0  056764          13223   13239   13249#
QDATI1  056766          13250#
QDATI2  056770          13251#
QDATI3  056772          13252#
QDATO0  056754          13218   13222   13244#
QDATO1  056756          13245#
QDATO2  056760          13246#
QDATO3  056762          13247#
QDONE   056774          13232   13254#
```

```
QPAT10  056734          13210   13234#
QPAT11  056736          13235#
QPAT12  056740          13236#
QPAT13  056742          13237#
QPAT20  056744          13213   13229   13239#
QPAT21  056746          13240#
QPAT22  056750          13241#
QPAT23  056752          13242#
QQBDON  106452          19341   19357#
QQBTP1  106420          19318   19323   19331   19337   19344#
QQBTP2  106440          19319   19332   19350#
QQB10   106450          19335   19338   19340   19355#
QQB2    106354          19328#
QQCDON  115246          21095   21104#
QQCTB0  115224          21078   21098#
QQCTB1  115230          21086   21091   21093   21099#
QQCTB2  115240          21085   21089   21093   21101#
QQC10   115244          21083   21090   21092   21094   21102#
QQC2    115172          21088#
QQQBF0  101142          17795   17825#  17839
QQQBF1  101156          17805   17810   17814   17827#  17848   17851   17854
QQQDON  101310          17860   17864#
QQQTP1  101172          17802   17815   17829#  17846   17855
QQQ10   101202          17817   17819   17821   17823   17834#
QQQ2    101042          17797#  17798
QQQ20   101204          17824   17839#
QQQ22   101216          17842#  17843
QQQ23   101250          17849   17850#
QQQ24   101262          17852   17854#
QQQ25   101306          17858   17862#
QQQ3    101072          17808#
QQQ4    101104          17812   17814#
Q1      056646          13207#
Q2      056662          13215#  13216
Q3    = 056664          13216#
Q4      056666          13218#
Q5      056710          13224#  13227
RBUF    120240          21875#  21984   22174   22208   22241   22270   22301   22317   22363   22366   22372   22386   23357
                        23362
RBUF2   124104          22654#  22763   22972   22999   23009   23037   23063   23092   23107   23155   23158   23164   23179
                        23188   23204   23207   23210   23225   23374   23379
RCSR    120236          21874#  21971   22052   22056*  22057   22060*  22061   22065*  22067   22184   22193   22206   22210
                        22221*  22227   22235*  22238*  22253   22255*  22260*  22284   22287*  22294*  22311*  22314   22320*
                        22335*  22338   22341*  22344*  22384   23334*
RCSR2   124102          22653#  22750   22841   22845*  22846   22849*  22850   22858*  22860   22981   22993   23007   23010
                        23021*  23026   23032*  23035*  23049   23051*  23056*  23077   23079*  23086*  23103*  23105   23110*
                        23130*  23132   23135*  23138*  23177   23182   23201   23220   23337*
RECCT1  127764          23319*  23356*  23394   23425#
RECCT2  127766          23320*  23373*  23405   23426#
REC1    127442          23324   23356#
REC2    127524          23328   23373#
REGR1   025542           6954    6957#
REGR2   025670           6982    6985#
REGR23  027764           7397    7400#
REGR3   026016           7010    7013#
REGR4   026142           7038    7041#
```

```
REGR5   026272            7068    7071#
REG01   027604            7362#   7474
REG1    003362            1178#   1181
REG1A   003410            1193#   1196
REG1E   003374            1179    1183#
REG2    003442            1208#   1211
REG2B   003470            1223#   1226
REG2E   003454            1209    1213#
REG3    003522            1239#   1242
REG3A   003550            1254#   1257
REG3E   003534            1240    1244#
REG4    003602            1270#   1273
REG4A   003630            1285#   1288
REG4E   003614            1271    1275#
REG45   030140            7433    7436#
REG5    003662            1302#   1305
REG5A   003710            1317#   1320
REG5E   003674            1303    1307#
REG6    003742            1333#   1336
REG6A   003770            1348#   1351
REG6E   003754            1334    1338#
RESET2  024164            6600    6608#
RESET3  024162            6606#
RESI    017416            5487#
RESTRT  001206             656#   23461   23479
RET     025072            6793    6801#
RETA    050310           11461   11463#
RETAT   022644            6308    6314#
RETBT   022674            6320    6326#
RETCT   022730            6334    6339#
RETR1   024376            6655    6658#
RETR2   024434            6670    6674#
RETR3   024460            6687    6691#
RET1    025102            6802    6804#
RET2    025114            6805    6807#
RET3    025124            6808    6811#
RET4    025122            6809#
RITSH   030324            7398    7434    7472    7475#
ROL1    015036            4768    4769    4770    4772#
ROL2    015040            4771    4774#
ROL3    015056            4777    4778    4779    4781#
ROL4    015060            4780    4783#
ROL5    015074            4786    4787    4789#
ROL6    015076            4788    4791#
ROL7    015120            4794    4795    4796    4799#
RORB1   015476            4939    4941#
RORB10  015560            4965    4966    4968#
RORB11  015562            4967    4970#
RORB12  015574            4971    4973#
RORB13  015606            4974    4975    4976    4978#
RORB14  015610            4977    4980#
RORB2   015510            4942    4943    4944    4946#
RORB3   015512            4945    4948#
RORB4   015522            4949    4950    4952#
RORB5   015524            4951    4954#
RORB7   015534            4955    4957#
```

ROR1	015144	4809	4810	4811	4813#							
ROR2	015146	4812	4815#									
ROR3	015164	4818	4819	4820	4822#							
ROR4	015166	4821	4824#									
ROR5	015204	4827	4828	4829	4831#							
ROR6	015206	4830	4833#									
ROR7	015222	4836	4837	4839#								
ROTX	012120	3751	3753*	3757	3763#	3776*	3778*	3780	3796*	3797	3801	
ROTXAD	012210	3797*	3799*	3805#								
ROT0A	011424	3569	3570	3573#								
ROT0B	011426	3572	3575#									
ROT0C	011450	3578	3579	3582#								
ROT1A	011500	3602	3603	3606#								
ROT1B	011502	3605	3608#									
ROT1C	011526	3611	3612	3615#								
ROT1D	011530	3614	3617#									
ROT1E	011560	3622	3623	3626#								
ROT2A	011614	3644	3646	3650#								
ROT2B	011616	3649	3652#									
ROT2C	011646	3656	3658	3661#								
ROT2D	011650	3660	3663#									
ROT2E	011704	3668	3670	3674#								
ROT3A	011734	3691	3694#									
ROT3B	011736	3693	3696#									
ROT3C	011764	3699	3702#									
ROT3D	011766	3701	3704#									
ROT3E	012014	3707	3710#									
ROT4	012052	3729	3731	3734#								
ROT5	012116	3756	3758	3761#								
ROT6	012150	3779	3782#									
ROT7	012206	3800	3803#									
RPSW	120250	21879#	22278	22280*	22296*	23325*						
RPSW2	124114	22658#	23072	23074*	23088*	23329*						
RRBDON	106612	19393	19408#									
RRBTP1	106566	19369	19375	19376*	19383	19396#	19404					
RRBTP2	106576	19370	19384	19400#								
RRBTP3	106606	19374	19389	19404#								
RRB10	106610	19387	19390	19392	19406#							
RRB2	106522	19380#										
RRCDON	115372	21136	21145#									
RRCTB0	115350	21119	21139#									
RRCTB1	115354	21127	21132	21134	21140#							
RRCTB2	115364	21126	21130	21134	21142#							
RRC10	115370	21124	21131	21133	21135	21143#						
RRC2	115316	21129#										
RRRDON	101462	17908	17926#									
RRREXP	101452	17899	17922#									
RRRTP1	101432	17877	17912#									
RRRTP2	101442	17883	17917#									
RRR10	101430	17885	17903	17906	17909#							
RRR2	101362	17894#										
RRR3	101364	17876	17895#	17900								
RRR4	101410	17902#	17904									
RTCPSW	120262	21886#	22486	22488*	22518*	22534	22536*	22555*	23331*			
RTCVT	120260	21885#	22424	22425*	22449*	22485	22487*	22498*	22506*	22517*	22533	22535*	22547*	22554*
		22568	22569*	22581*	22589	22590*	22603*	23316	23330*	23420*		

```
RTI1    023100          6387    6391#
RTI2    023104          6388    6393#
RTRAP = 000010           534#    756*   757*   771*   772*   6709*
RTRAP1= 000034           525#    785*   786*   800*   801*
RTRAP2= 000020           524#    822*   823*   837*   838*
RTRAP3= 000030           523#    851*   852*   866*   867*
RTRAP4= 000014           522#    888*   889*   903*   904*   6308*  6320*  6334*
RTRAP5= 000004           521#    923*   924*   938*   939*   952*   953*   967*   968*   6655*  6670*  6687*  6707*
RTS1    013400          4207    4212#
RTT1    022770          6352    6355#
RTT2    022776          6353    6359#
RTT3    023030          6367    6371#
RTT4    023040          6368    6375#
RTT5    023002          6365#   6373    6378
RTT6    023052          6376    6380#
RVECT   120246         21878#  22049   22050*  22070*  22223   22224*  22234*  22240*  22249   22250*  22261*  22269*  22277
                       22279*  22291*  22295*  22300*  22309   22310*  22324*  22333   22334*  22348*  23315   23324*  23419*
RVECT2  124112         22657#  22838   22839*  22863*  23022   23023*  23031*  23038*  23046   23047*  23057*  23064*  23071
                       23073*  23083*  23087*  23093*  23101   23102*  23114*  23128   23129*  23142*  23313   23328*  23417*
ROTRAP  023312          6456    6461#
R1ERR   003426          1194    1199#
R2ERR   003506          1224    1229#
R3ERR   003566          1255    1260#
R4ERR   003646          1286    1291#
R5ERR   003726          1318    1323#
R6ERR   004006          1349    1354#
SADR    055300         12718*  12729   12734*  12737#
SBC1    014730          4716    4717    4718    4720#
SBC2    014732          4719    4722#
SBC3    014750          4725    4726    4727    4729#
SBC4    014752          4728    4731#
SBC5    014770          4734    4735    4736    4738#
SBC6    014772          4737    4740#
SBC7    015012          4744    4745    4747#
SB0     012224          3821    3823#
SB2     012304          3861    3863#
SB4     012366          3901    3903#
SB5     012430          3929#
SB5A    012422          3925    3927#
SB5X    012432          3922*   3924    3931#   3932
SB5XAD  012434          3921    3927    3932#
SB6     012464          3953#
SB6X    012466          3948*   3949    3955#
SB7     012516          3977#
SB7X    012520          3972*   3979#   3980
SB7XAD  012522          3973    3980#
SCOPE = 000240           504#
SC3     020262          5685*   5704#   5713*   5714
SC4     020276          5686*   5708#   5712*
SDATO0  055314         12676   12679   12702   12705   12744#
SDATO1  055316         12745#
SDATO2  055320         12746#
SDATO3  055322         12747#
SDONE   055324         12734   12749#
SECND   124062         22644#
SERRO   055210         12663   12715#
```

```
SERR1   055116          12668   12686#  12723   12728
SERR10  055226          12719#  12735
SERR4   055260          12693   12731#
SETBR   001410            707#    744
SETCC   001430            711#    739
SETCD   020260           5700    5703#   5715
SETREG  050012           9681    9720   11366#
SETUP   001372            704#
SET2BR  001500            716     720#
SHL     003174           1080#   1083
SHLE    003210           1081    1085#
SHR     003254           1106#   1109
SHRE    003270           1107    1111#
SKTST2  024202           6611#
SLU1ST  120264           6507    6521   11499   21836   21888#
SLU2ST  124122          22403   22614   22640   22662#
SL1HLT  122670            674*  22398#
SL2HLT  126670            675*  23241#
SNMB0A  005676           2189    2190    2192#
SNMB1A  005746           2236    2237    2239#
SNMB1B  005750           2238    2241#
SNMB1C  005772           2246    2247    2248    2250#
SNMB2A  006052           2298    2299    2301#
SNMB2B  006054           2300    2303#
SNMB2C  006062           2304    2306#
SNMB2D  006102           2310    2311    2312    2314#
SNMB2E  006104           2313    2316#
SNMB3A  006204           2372    2373    2375#
SNMB3B  006206           2374    2377#
SNMB3C  006224           2380    2381    2382    2384#
SNMB3D  006226           2383    2386#
SNM0A   005654           2165    2166    2167    2169#
SNM1A   005722           2212    2213    2214    2216#
SNM2A   006016           2269    2270    2271    2273#
SNM2B   006020           2272    2275#
SNM3A   006142           2339    2340    2341    2343#
SNM3B   006144           2342    2345#
SNM4A   006260           2408    2409    2411#
SNM4B   006262           2410    2413#
SNM5A   006316           2436    2437    2439#
SNM5B   006320           2438    2441#
SNM6A   006356           2464    2465    2467#
SNM6B   006360           2466    2469#
SNM7A   006414           2491    2492    2494#
SNM7B   006416           2493    2496#
SOB1    016150           5135#   5142
SOB2    016156           5135    5136    5138#
SOB3    016160           5137    5140#
SOB4    016200           5143    5144    5145    5148#
SOPA    005610           2138    2140#
SOPB    005630           2142    2145#
SOPB0A  004204           1496    1498#
SOPB0B  004214           1499    1502#
SOPB1A  004256           1544    1546#
SOPB1B  004274           1547    1550    1553#
SOPB1C  004314           1575    1577#
```

```
SOPB1D  004334  1582  1585#
SOPB2A  004420  1641  1643#
SOPB2B  004440  1647  1651#
SOPB2C  004464  1670  1672#
SOPB2D  004510  1678  1682#
SOPB3A  004774  1831  1833#
SOPB3B  005020  1836  1838  1843#
SOPB3C  005042  1869  1871#
SOPB3D  005064  1877  1880#
SOPX    005602  2137#  2140*  2141*  2147
SOPXAD  005632  2143*  2147#
SOPZA   004356  1608  1610#
SOP0A   004120  1443  1445#
SOP0B   004132  1448  1450#
SOP0C   004150  1470  1472#
SOP0D   004174  1474  1477  1482#
SOP1A   004226  1519  1521#
SOP1B   004240  1522  1526#
SOP2B   004376  1613  1618#
SOP3A   004734  1797  1799#
SOP3B   004750  1802  1805#
SOP4A   005204  1933  1935#
SOP4B   005224  1938  1943#
SOP5A   005256  1974  1976#
SOP5B   005272  1979  1982#
SOP6A   005312  2001  2003#
SOP6B   005326  2004  2007#
SOP7A   005350  2028  2030#
SOP7B   005364  2031  2034#
SPAT10  055304  12660  12680  12691  12706  12739#
SPAT11  055306  12740#
SPAT12  055310  12741#
SPAT13  055312  12742#
SR0  = 177572  9192#  9272*  9381  9442  9489*  9491  9497*  9817*  9819*  9841*  9843*  9865*  9867*
               9890*  9892*  9911*  9913*  9941*  9943*  9965*  9967*  9989*  9991*  10013*  10015*  10055*
               10098*  10214  10215*  10286  10301*  10344  10359*  10431  10447*  10450*  10525  10541*  10544*
               10614*  10654  10659  10675*  10676  10690*  10772  10788*  10813  10815*  11316*
SR1  = 177574  9193#  9505
SR2  = 177576  9194#  9451  9471  10287  10345  10432  10526  10607  10634  10642  10655  10660  10680
               10691  10773  10814
SR3  = 172516  9195#  9511*  9512  9519*  9520  9816*  9840*  9864*  9889*  9910*  9940*  9964*  9988*
               10012*  10054*
SSBDON  106754  19443  19458#
SSBTP1  106730  19419  19425  19426*  19433  19446#  19454
SSBTP2  106740  19420  19434  19450#
SSBTP3  106750  19424  19439  19454#
SSB10   106752  19437  19440  19442  19456#
SSB2    106664  19430#
SSCDON  115516  21178  21186#
SSCTB0  115500  21159  21181#
SSCTB1  115504  21167  21172  21174  21176  21182#
SSC10   115514  21164  21171  21173  21175  21177  21184#
SSC2    115436  21169#
SSSA1   101624  17948  17961  17973#
SSSBF0  101614  17939  17952  17954  17969#
SSSDON  101646  17966  17986#
```

```
SSSTP1  101634          17945  17955  17979#
SSS10   101644          17947  17953  17958  17964  17984#
SSS2    101534          17950#
START   001200            647   655#
STATUS= 177776            532#  758*  762   773*  776   787*  791   802*  805   824*  828   839*  842
                          853*  857   868*  871   890*  894   905*  908   925*  929   940*  943   954*
                          958   969*  972   6461*  6462  6489*  6490  6514*  6520*  6534*  6541*  6555*  6608*
                         6634*  6637
STBOT   001000            538   539   644#  648   657   664   755   764   766   768   778   784   793
                          795   797   807   821   330   832   834   844   850   859   861   863   873
                          887   896   898   900   910   922   931   933   935   945   951   960   962
                          964   974  1375  4099  4114  4129  4135  4205  5161  5174  5185  5536  5538
                         5549  5558  5571  5578  5886  6307  6319  6326  6333  6339  6349  6365  6385
                         6403  6422  6428  6451  6465  6478  6493  6502  6533  6554  6599  6623  6640
                         6654  6658  6725  6741  6746  6794  6801  6804  6807  6906  9268  11496  21769
                        21851 23474
STCDFS  103414          18379  18398  18417  18436  18455  18505#
STCDT   103544          18511  18517  18524  18540#
STCFDS  102762          18199  18218  18237  18256  18275  18329#
STCFT   103112          18335  18341  18348  18364#
STCIBF  117100          21510  21516  21523  21541#
STCSUB  116752          21298  21308  21318  21328  21339  21349  21359  21369  21380  21390  21400  21410  21420
                        21430  21440  21451  21462  21505#  21559
STP     022602            6298   6301#
STP3    023236            6430#
STP3D   023240            6429   6432#
STP4    024340            6642#
STP4E   024342            6620   6641   6644#
STXBF   117474           21673*  21674*  21681  21684  21691  21697#
STXSUB  117362           21582  21591  21600  21610  21620  21630  21671#
SUB0    006500            2546   2547   2548   2550#
SUB0A   006502            2549   2552#
SUB1    014616            4675   4676   4678#
SUB2    014620            4677   4680#
SUB3    014642            4684   4685   4687#
SUB4    014644            4686   4689#
SUB5    014662            4693   4694   4696#
SUB6    014664            4695   4698#
SUB7    014704            4701   4702   4703   4705#
SWB1    014114            4455   4456   4458#
SWB2    014116            4457   4460#
SWB3    014134            4463   4464   4465   4467#
SWR     021264            6011#  6015*  6763*  6778*  6785  9273  11497  21888  22401  22662  23255
SWREG   000176             632#
SXT0    016016            5062   5063   5064   5065   5068#
SXT1    016020            5067   5070#
SXT2    016050            5076   5077   5078   5079   5082#
SX10    055152           12700#
SX11    055176           12708#  12711
SX12    055204           12709   12711#
SX2     055054           12671#
SX3     055056           12672#  12715
SX4     055062           12674#
SX5     055106           12682#  12685
SX6     055114           12683   12685#
SX7     055120           12690#  12718
```

```
SX8     055144          12697#
SX9     055146          12698#  12731
S1      025312           6882#  6883   7170   7211   7743   7754   7800
S10     025334           6891#  8957   8987
S11     025336           6892#
S2      025314           6883#  7171   7194   7744   7777
S3      025316           6884#
S4      025320           6885#
S5      025322           6886#  6887   8379   8388   8430
S6      025324           6887#  8380   8409
S7      025326           6888#
S8      025330           6889#
S9      025332           6890#  6891   8956   8966   9008
TABLE   025136           6780   6815#
TBIT  = 000020           9244#  11329  11334  11347
TBITPS  037236           9259#  9271*  11332* 11347  11349  11350*
TBLEND  126712          23252#  23286
TBL1    011216           3467   3468*  3469*  3470*  3471*  3472   3481#  3497*  3498*  3499*  3500*  3501   3505
                         3506   3507   3508   3509   3524   3525*  3526*  3527*  3528*  3529   3547*  3548*  3549*
                         3550*  3551
TBL2    011264           3496   3509#  3546
TBUF    120244          21877#  21915  21931* 21936* 21953* 21956* 22160* 22163* 22183* 22205* 22226* 22252* 22283*
                        22313*  22337* 22357* 22383* 23350*
TBUF2   124110          22656#  22689  22705* 22710* 22732* 22735* 22958* 22961* 22980* 23006* 23025* 23048* 23076*
                        23104*  23131* 23150* 23176* 23200* 23218* 23367*
TCCBF0  111466          20204   20212  20222#
TCCBF1  111476          20208   20224#
TCCDON  111506          20217   20227#
TCC2    111426          20205#
TCC3    111456          20213   20216#
TCSR    120242          21876#  21901  21932  21937  21941  21954  21959  21999  22002* 22003  22006* 22007  22010*
                        22012   22024  22028* 22029  22032* 22033  22037* 22039  22078* 22081  22088* 22093* 22102*
                        22106   22108* 22113* 22127* 22133  22135* 22144* 22155* 22159* 22161  22166* 22181* 22188*
                        22204*  22209* 22220* 22222* 22229* 22232* 22251* 22258* 22264* 22267* 22282* 22286* 22312*
                        22316*  22336* 22355* 22358  22362* 22381* 22389* 22393* 23332* 23333* 23353* 23384*
TCSR2   124106          22655#  22675  22706  22711  22715  22733  22738  22778  22781* 22782  22785* 22786  22795*
                        22797   22809  22813* 22814  22817* 22818  22826* 22828  22871* 22874  22881* 22886* 22895*
                        22899   22901* 22906* 22920* 22926  22928* 22937* 22953* 22957* 22959  22964* 23020* 23151
                        23180*  23199* 23203* 23336* 23370* 23385*
TDATI0  052310          11961   11966  11981  11991  11996  12022  12046#
TDATI1  052312          12047#
TDATI2  052314          12019*  12048#
TDATI3  052316          12020*  12049#
TDATO0  052300          11977   11980  12016  12021  12041#
TDATO1  052302          12042#
TDATO2  052304          12043#
TDATO3  052306          12044#
TDEC1   022102           6167   6170#
TDEC2   022116           6177   6179#
TDEC3   022146           6189   6191#
TDEC4   022200           6196   6199#
TDEC6   022214           6202#
TDEC7   022216           6200   6204#
TDEC77  023562           6516   6525#
TDEC8   023560           6518   6523#
TDONE   052320          12028   12051#
```

```
TEMP1   025276  6517* 6528  6535* 6548  6558* 6570  6624* 6644  6872# 6873  6907* 6927  6957
                6985  7013  7041  7072  7102* 7108* 7123* 7130* 7148* 7161* 7355* 7363  7400
                7437  7476* 7484* 7498* 7506* 7513* 7518* 7524* 7555* 7562*
TEMP2   025300  6874# 6875  6908* 6930  6934  6951* 6960  6963  6979* 6988  6991  7007* 7016
                7019  7035* 7045  7048  7065* 7075  7078  7095* 7103* 7109* 7115* 7119* 7124*
                7131* 7136* 7142* 7149* 7155* 7356* 7364  7401  7438  7477* 7525* 7532* 7543*
                7548* 7556* 7563*
TEMP3   025302  6876# 6877  6909* 6940  6952* 6969  6980* 6997  7008* 7025  7036* 7054  7066*
                7084  7096* 7104* 7110* 7125* 7132* 7137* 7143* 7150* 7156* 7357* 7368  7371
                7392* 7405  7408  7428* 7442  7445  7466* 7478* 7490* 7494* 7499* 7519* 7526*
                7537* 7544* 7549* 7564*
TEMP4   025304  6878# 6879  6910* 6936  6965  6993  7021  7050  7080  7111* 7126* 7138* 7144*
                7151* 7157* 7162* 7358* 7377  7395* 7414  7431* 7451  7469* 7479* 7485* 7500*
                7507* 7533* 7538* 7550* 7565*
TEMP5   025306  6880# 7359* 7380  7394* 7417  7430* 7454  7468* 7480* 7501* 7508* 7514* 7527*
                7551* 7557*
TEMP6   025310  6881# 7360* 7373  7410  7447  7486* 7502* 7509* 7520* 7528* 7539* 7558* 7566*
TENSAV  024722  6755* 6768# 6779
TESTN1  021266  5934  6013#
TEST1   014052  4432  4433  4434  4436#
TEST2   014054  4435  4438#
TEST3   014072  4442  4443  4445#
THRPRT  025344  6798  6901#
TICKER  127376  23330 23346#
TICKS   127770  23321* 23346* 23381 23427#
TKB   = 177562   528#  6583
TOFF    047724  10110 10160  10211 10709 10798 11329#
TOLER   124050  22637 22639#
TON     047760  10153 10203  10238 10734 10852 11347#
TONT1   023162   6407  6416#
TO10    021220    609   991  5980#  6215  6727  6747  6797  21849
TO114   021232    621  5990#
TO14    021222    611   992  5982#  6257  6432  6610
TO20    021224    613  5984#  6225
TO244   021234    624  5992#
TO250   021236    626  5994#  9269 10096 10314 10366 10456 10550 10623 10701 10790 10850 10995
                11136 11257
TO30    021226    616  5986#
TO34    021230    618   993  5988#  6247  6569
TO4     021216    607   662   990  5978#  6267  6280  6301  6497  6506  6508  6526  6726 10237
                10733 10789 10848 21848
TPAT10  052260  11960 11990  12031#
TPAT11  052262  12032#
TPAT12  052264  12033#
TPAT13  052266  12034#
TPAT20  052270  11969 11999  12036#
TPAT21  052272  12037#
TPAT22  052274  12038#
TPAT23  052276  12039#
TPB   = 177566    509#   529#  6629* 6632* 21792* 21795* 21806* 21809* 23486*
TPS   = 177564    508#
TPSW    120254  21881# 22129  22131* 22149* 23323*
TPSW2   124120  22660# 22922  22924* 22942* 23327*
TRACE   023156   6404  6412#
TRAPA = 000077    533#   759   774  6712
TRAPB   023434   6466  6484  6497#
```

```
TRAPC = 104777            541#    803
TRAPPC  037226           9255#
TRAPPS  037230           9256#
TRAP10  024724           6756    6769#
TRCSR = 177560            527#   6584*   6586
TRC1    023226           6423    6426    6428#
TRPADR  023310           6458#   6465
TRT   = 000003            519#    891     906    6255
TR0     023672           6536    6543#
TR2     023676           6539    6547#
TR3     024004           6557    6564#
TR4     024010           6559    6567#
TR5     024006           6561    6565#
TSTSPC  037166           9173#
TST160  030360           7399    7435    7473    7482#
TST161  030406           7483    7488#
TST162  030424           7489    7492#
TST163  030442           7493    7496#
TST164  030504           7497    7504#
TST165  030542           7505    7511#
TST166  030566           7512    7516#
TST167  030616           7517    7522#
TST170  030654           7523    7530#
TST171  030700           7531    7535#
TST172  030726           7536    7541#
TST173  030750           7542    7546#
TST174  031006           7547    7553#
TST175  031042           7554    7560#
TST176  031114           7578#
TST177  031162           7598#
TST200  031230           7618#
TST201  031302           7640#
TST202  031350           7660#
TST203  031416           7680#
TST204  031470           7702#
TST205  031536           7722#
TST206  031630           7750#
TST207  031706           7773#
TST210  031764           7796#
TST211  032042           7819#
TST212  032116           7842#
TST213  032172           7865#
TST214  032246           7888#
TST215  032324           7911#
TST216  032402           7934#
TST217  032456           7957#
TST220  032532           7994#
TST221  032602           8015#
TST222  032652           8036#
TST223  032722           8057#
TST224  032776           8080#
TST225  033046           8101#
TST226  033116           8122#
TST227  033172           8145#
TST230  033242           8166#
TST231  033312           8187#
```

TST232	033362	8208#						
TST233	033436	8231#						
TST234	033506	8252#						
TST235	033556	8273#						
TST236	033626	8294#						
TST237	033702	8317#						
TST240	033752	8338#						
TST241	034022	8359#						
TST242	034112	8386#						
TST243	034162	8407#						
TST244	034232	8428#						
TST245	034302	8449#						
TST246	034350	8470#						
TST247	034416	8491#						
TST250	034464	8512#						
TST251	034534	8533#						
TST252	034604	8554#						
TST253	034652	8575#						
TST254	034720	8602#						
TST255	034774	8623#						
TST256	035050	8644#						
TST257	035130	8667#						
TST260	035204	8688#						
TST261	035260	8709#						
TST262	035340	8732#						
TST263	035414	8753#						
TST264	035470	8774#						
TST265	035544	8795#						
TST266	035624	8818#						
TST267	035700	8839#						
TST270	035754	8860#						
TST271	036034	8883#						
TST272	036110	8904#						
TST273	036164	8925#						
TST274	036232	8943#						
TST275	036310	8963#						
TST276	036364	8984#						
TST277	036440	9005#						
TST300	036514	9026#						
TST301	036566	9047#						
TST302	036640	9068#						
TST303	036712	9089#						
TST304	036766	9110#						
TST305	037042	9131#						
TST306	037114	9152#						
TST37	026454	6956	6984	7012	7040	7070	7100	7106#
TST40	026510	7107	7113#					
TST41	026524	7114	7117#					
TST42	026542	7118	7121#					
TST43	026576	7122	7128#					
TST44	026630	7129	7134#					
TST45	026662	7135	7140#					
TST46	026714	7141	7146#					
TST47	026752	7147	7153#					
TST50	027002	7154	7159#					
TST51	027054	7176#						

TST52	027114	7193#	
TST53	027154	7210#	
TST54	027214	7227#	
TST55	027252	7244#	
TST56	027310	7261#	
TST57	027346	7278#	
TST60	027406	7295#	
TST61	027446	7312#	
TST62	027504	7329#	
TS1	001372	669	703#
TS10	002614	950#	
TS100	005656	2168	2183#
TS101	005700	2191	2206#
TS102	005724	2215	2229#
TS103	005774	2249	2263#
TS104	006030	2277	2291#
TS105	006114	2318	2331#
TS106	006154	2347	2362#
TS107	006236	2388	2401#
TS11	003002	999#	
TS110	006270	2414	2427#
TS111	006330	2443	2456#
TS112	006366	2470	2483#
TS113	006424	2497	2509#
TS114	006442	2515	2525#
TS115	006460	2531	2541#
TS116	006510	2553	2567#
TS117	006604	2596	2607#
TS12	003020	1003	1009#
TS120	006650	2622	2633#
TS121	006726	2655	2665#
TS122	006766	2680	2693#
TS123	007006	2700	2712#
TS124	007026	2719	2732#
TS125	007046	2739	2756#
TS126	007102	2769	2782#
TS127	007124	2790	2803#
TS13	003040	1013	1019#
TS130	007152	2813	2828#
TS131	007202	2837	2849#
TS132	007234	2861	2874#
TS133	007260	2879	2891#
TS134	007304	2896	2908#
TS135	007330	2913	2919#
TS136	007362	2931	2937#
TS137	007414	2949	2955#
TS14	003060	1023	1029#
TS140	007460	2971	2977#
TS141	007554	3004	3011#
TS142	007654	3039	3045#
TS143	007730	3062	3068#
TS144	010024	3093	3099#
TS145	010102	3116	3122#
TS146	010156	3138	3144#
TS147	010234	3160	3172#
TS15	003100	1032	1052#

```
TS150   010264          3184    3196#
TS151   010326          3213    3224#
TS152   010414          3246    3257#
TS153   010524          3279    3292#
TS154   010566          3306    3320#
TS155   010654          3341    3356#
TS156   010750          3378    3392#
TS157 --011052          3414    3427#
TS16    003126          1057    1063#
TS160   011152          3448    3466#
TS161   011220          3473    3495#
TS162   011266          3502    3523#
TS163   011334          3530    3545#
TS164   011402          3552    3565#
TS165   011452          3581    3597#
TS166   011562          3625    3639#
TS167   011706          3673    3687#
TS17    003160          1070    1076#
TS170   012016          3709    3724#
TS171   012054          3733    3750#
TS172   012122          3760    3775#
TS173   012152          3781    3795#
TS174   012212          3802    3818#
TS175   012234          3824    3837#
TS176   012260          3842    3856#
TS177   012314          3864    3878#
TS2     001614           754#
TS20    003212          1084    1091#
TS200   012340          3882    3896#
TS201   012374          3904    3920#
TS202   012436          3928    3947#
TS203   012470          3952    3971#
TS204   012524          3976    4017#
TS205   012772          4072    4095#
TS206   013360          4189    4204#
TS207   013410          4213    4232#
TS21    003240          1096    1102#
TS210   013450          4247    4253#
TS211   013514          4269    4275#
TS212   013560          4290    4296#
TS213   013626          4313    4331#
TS214   013712          4357    4364#
TS215   014014          4397    4414#
TS216   014034          4421    4428#
TS217   014074          4444    4450#
TS22    003272          1110    1137#
TS220   014136          4466    4483#
TS221   014254          4520    4527#
TS222   014340          4552    4569#
TS223   014432          4597    4604#
TS224   014552          4638    4645#
TS225   014574          4652    4670#
TS226   014706          4704    4711#
TS227   015014          4746    4763#
TS23    003304          1141    1147#
TS230   015122          4798    4804#
```

TS231 015224 4838 4844#
TS232 015334 4880 4886#
TS233 015454 4925 4933#
TS234 015622 4981 4988#
TS235 015670 5005 5013#
TS236 015770 5043 5057#
TS237 016052 5081 5095#
TS24 003320 1150 1156#
TS240 016140 5118 5131#
TS241 016202 5147 5160#
TS242 016324 5188 5204#
TS243 016370 5218 5225#
TS244 016426 5234 5240#
TS245 016472 5250 5256#
TS246 016532 5265 5271#
TS247 016570 5280 5286#
TS25 003334 1159 1165#
TS250 016634 5295 5301#
TS251 016700 5310 5324#
TS252 016746 5336 5342#
TS253 017016 5357 5363#
TS254 017072 5378 5384#
TS255 017146 5399 5405#
TS256 017222 5420 5426#
TS257 017300 5441 5447#
TS26 003350 1168 1174#
TS260 017356 5462 5476#
TS261 017416 5480 5485 5497#
TS262 017456 5519#
TS263 017570 5548#
TS264 017650 5566#
TS265 017732 5579 5594#
TS266 020064 5636#
TS267 020122 5655#
TS27 003376 1182 1189#
TS270 020144 5662 5682#
TS271 020336 5731#
TS272 020372 5743 5749#
TS273 020426 5761 5767#
TS274 020516 5788 5794#
TS275 020606 5816 5829#
TS276 020624 5833 5839#
TS277 020646 5843 5849#
TS3 001736 783#
TS30 003430 1198 1204#
TS300 020676 5854 5860#
TS301 020722 5864 5870#
TS302 020744 5874 5884#
TS303 020762 5889 5899#
TS304 021012 5905 5911#
TS305 021054 5919 5925#
TS306 021304 6020#
TS307 021504 6076 6081#
TS31 003456 1212 1219#
TS310 021744 6124 6129#
TS311 022064 6158 6165#

TS312	022102	6175#		
TS313	022126	6180	6186#	
TS314	022216	6209#		
TS315	022252	6219#		
TS316	022306	6230#		
TS317	022342	6241#		
TS32	003510	1228	1235#	
TS320	022376	6251#		
TS321	022432	6261#		
TS322	022476	6269	6274#	
TS323	022532	6285#		
TS324	022614	6306#		
TS325	022644	6318#		
TS326	022704	6327	6332#	
TS327	022742	6340	6347#	
TS33	003536	1243	1250#	
TS330	022776	6356	6363#	
TS331	023052	6384#		
TS332	023112	6395	6401#	
TS333	023176	6421#		
TS334	023246	6447#		
TS335	023506	6513#		
TS336	023606	6532#		
TS337	023716	6553#		
TS34	003570	1259	1266#	
TS340	024050	6577#		
TS341	024114	6581	6587	6593#
TS342	024202	6597	6616#	
TS343	024356	6649#		
TS344	024410	6659	6668#	
TS345	024440	6675	6686#	
TS346	024470	6692	6706#	
TS347	024606	6739#		
TS35	003616	1274	1281#	
TS350	024644	6752#		
TS351	037334	9284#		
TS352	037370	9303#		
TS353	037432	9322#		
TS354	037520	9341	9350#	
TS355	037556	9357	9380#	
TS356	037602	9392#		
TS357	037622	9404#		
TS36	003650	1290	1298#	
TS360	037642	9416#		
TS361	037662	9428#		
TS362	037702	9440#		
TS363	040020	9486#		
TS364	040054	9503#		
TS365	040136	9522	9532#	
TS366	040226	9559	9566#	
TS367	040324	9595	9604#	
TS37	003676	1306	1313#	
TS370	040430	9634	9641#	
TS371	040534	9670	9677#	
TS372	040700	9712#		
TS373	041064	9718	9795#	

TS374	041540	9918	9930#	
TS375	042070	10034#		
TS376	042436	10097	10108#	
TS377	042612	10158#		
TS4	002100	820#		
TS40	003730	1322	1329#	
TS400	043000	10209#		
TS401	043122	10255#		
TS402	043400	10274	10320#	
TS403	043570	10402#		
TS404	044070	10425	10458	10498#
TS405	044354	10519	10552	10593#
TS406	044530	10630#		
TS407	045104	10708#		
TS41	003756	1337	1344#	
TS410	045242	10739#		
TS411	045306	10750	10760#	
TS412	045440	10797#		
TS413	045744	10864#		
TS414	046440	11001#		
TS415	047172	11062	11143#	
TS416	047572	11263#		
TS417	047654	11294#		
TS42	004010	1353	1374#	
TS420	050364	11507#		
TS421	050456	11543#		
TS422	050512	11568#		
TS423	050742	11656#		
TS424	051110	11713#		
TS425	051156	11741#		
TS426	051516	11854#		
TS427	052020	11958#		
TS43	004032	1378	1384#	
TS430	052324	12063#		
TS431	054134	12460#		
TS432	055022	12658#		
TS433	055330	12759#		
TS434	055474	12824#		
TS435	055640	12886#		
TS436	056036	12959#		
TS437	056174	13020#		
TS44	004052	1387	1393#	
TS440	056350	13086#		
TS441	056524	13153#		
TS442	056646	13205#		
TS443	057000	13264#		
TS444	057566	13409#		
TS445	060142	13512#		
TS446	060536	13622#		
TS447	060766	13689#		
TS45	004072	1396	1402#	
TS450	061200	13755#		
TS451	061504	13855#		
TS452	062314	14031#		
TS453	063114	14203#		
TS454	064014	14395#		

```
TS455   064252          14470#
TS456   064476          14557#
TS457   065252          14730#
TS46    004112           1405    1441#
TS460   066254          14956#
TS461   067116          15161#
TS462   067722          15405#
TS463   070176          15503#
TS464   070730          15682#
TS465   071422          15837#
TS466   072154          16015#
TS467   072506          16136#
TS47    004140           1451    1467#
TS470   073010          16269#
TS471   073372          16414#
TS472   073724          16552#
TS473   074336          16703#
TS474   075272          16928#
TS475   076506          17194#
TS476   077146          17354#
TS477   077652          17516#
TS5     002222            849#
TS50    004176           1481    1494#
TS500   100550          17683#
TS501   100644          17701   17730#
TS502   101026          17790#
TS503   101314          17874#
TS504   101466          17936#
TS505   101652          17996#
TS506   102016          18049#
TS507   102162          18099#
TS51    004216           1501    1516#
TS510   102320          18145#
TS511   102474          18196#
TS512   103126          18376#
TS513   103560          18551#
TS514   103620          18578#
TS515   103740          18625#
TS516   104000          18650#
TS517   104040          18677#
TS52    004242           1525    1539#
TS520   104174          18733#
TS521   104332          18784#
TS522   104460          18831#
TS523   104616          18882#
TS524   104764          18933#
TS525   105132          18985#
TS526   105264          19031#
TS527   105434          19082#
TS53    004276           1552    1569#
TS530   105552          19125#
TS531   105710          19174#
TS532   106034          19222#
TS533   106164          19269#
TS534   106314          19317#
TS535   106456          19367#
```

TS536	106616	19418#	
TS537	106760	19467#	
TS54	004336	1584	1601#
TS540	107110	19514#	
TS541	107244	19560#	
TS542	107406	19612#	
TS543	107560	19665#	
TS544	110500	19933#	
TS545	110562	19969#	
TS546	110636	20006#	
TS547	110736	20037#	
TS55	004400	1617	1634#
TS550	111044	20070#	
TS551	111144	20102#	
TS552	111240	20134#	
TS553	111350	20171#	
TS554	111414	20200#	
TS555	111512	20239#	
TS556	112512	20483#	
TS557	113266	20656#	
TS56	004442	1650	1662#
TS560	114572	20959#	
TS561	114704	20998#	
TS562	115016	21038#	
TS563	115130	21076#	
TS564	115252	21116#	
TS565	115376	21156#	
TS566	115522	21197#	
TS567	115660	21238#	
TS57	004512	1681	1693#
TS570	115736	21266#	
TS571	116014	21295#	
TS572	117114	21555#	
TS573	117154	21579#	
TS574	117512	21710#	
TS575	117620	21751#	
TS576	117710	21781#	
TS577	120314	21898#	
TS6	002350	886#	
TS60	004564	1717	1722#
TS600	120342	21912#	
TS601	120370	21925#	
TS602	120456	21952#	
TS603	120510	21960	21968#
TS604	120536	21981#	
TS605	120564	21998#	
TS606	120660	22013	22020#
TS607	120776	22048#	
TS61	004644	1746	1751#
TS610	121114	22077#	
TS611	121206	22101#	
TS612	121306	22126#	
TS613	121424	22154#	
TS614	121532	22180#	
TS615	121602	22203#	
TS616	121644	22211	22219#

```
TS617   121776          22247#
TS62    004716           1771    1791#
TS620   122130          22276#
TS621   122270          22307#
TS622   122372          22331#
TS623   122474          22354#
TS624   122570          22378#
TS625   122722          22411#
TS626   122750          22423#
TS627   123104          22454#
TS63    004752           1804    1823#
TS630   123206          22459   22483#
TS631   123374          22527#
TS632   123530          22531   22561#
TS633   123634          22565   22587#
TS634   123732          22610#
TS635   124152          22672#
TS636   124200          22686#
TS637   124226          22699#
TS64    005022           1842    1862#
TS640   124310          22703   22726#
TS641   124360          22730   22739   22747#
TS642   124406          22760#
TS643   124434          22777#
TS644   124546          22793   22798   22805#
TS645   124702          22837#
TS646   125036          22870#
TS647   125130          22894#
TS65    005066           1879    1885#
TS650   125230          22919#
TS651   125346          22947#
TS652   125472          22951   22978#
TS653   125556          23005#
TS654   125604          23011   23019#
TS655   125714          23044#
TS656   126016          23070#
TS657   126142          23099#
TS66    005170           1914    1928#
TS660   126230          23121#
TS661   126342          23125   23148#
TS662   126422          23170#
TS663   126516          23174   23198#
TS664   126574          23215#
TS665   126744          23264#
TS666   127072          23268   23304#
TS67    005226           1942    1964#
TS7     002472            921#
TS70    005274           1981    1996#
TS71    005330           2006    2022#
TS72    005366           2033    2040#
TS73    005424           2055    2060#
TS74    005474           2080    2085#
TS75    005534           2100    2105#
TS76    005600           2121    2135#
TS77    005634           2144    2160#
TTBDON  107104          19491   19505#
```

```
TTBTP1  107062          19468   19473   19481   19487   19494#
TTBTP2  107072          19469   19482   19498#
TTB10   107102          19485   19488   19490   19503#
TTB2    107016          19478#
TTCDON  115654          21219   21228#
TTCSR = 177564            526#   6504    6519*   6525*   6537*   6549*   6556*   6572*   6622*   6627    6630    6633*   6645*
                         6753*  21793   21796*  21805*  21807   21810*  21828*  23484   23488
TTCTB0  115632          21199   21222#
TTCTB1  115636          21208   21215   21217   21223#
TTCTB2  115646          21207   21213   21225#
TTC10   115652          21204   21212   21214   21216   21218   21226#
TTC2    115570          21210#
TTTA1   101774          18029#
TTTA2   101776          18003*  18009   18013   18030#
TTTBF0  101762          17998   18003   18015   18024#  18030
TTTDON  102012          18021   18039#
TTTTP1  102000          18006   18016   18031#
TTT10   102010          18008   18014   18019   18036#
TTT2    101726          18011#
TTT3    101752          18018#  18020
TVECT   120252          21880#  22021   22022*  22042*  22079   22080*  22087*  22095*  22104   22105*  22114*  22120*  22128
                        22130*  22141*  22148*  22157   22158*  22170*  23314   23322*  23418*
TVECT2  124116          22659#  22806   22807*  22831*  22872   22873*  22880*  22888*  22897   22898*  22907*  22913*  22921
                        22923*  22934*  22941*  22955   22956*  22968*  23312   23326*  23416*
TYPE    131174          23446   23478   23482#
T105    052120          11984   11986#
T12     052122          11990#
T13     052136          11993#  11994
T14     052164          12004#
T15     052166          12005#
T16     052170          12006#
T17     052206          12011   12013#
T2      052036          11963#  11964
T20     052246          12024#  12027
T23     052254          12025   12027#
T3      052062          11973#
T4      052064          11974#
T5      052066          11975#
T6      052112          11983#  11986
UDONE   057566          13370   13403#
UERR0   057122          13273   13289#
UERR3   057454          13351   13363#
UFLAG   057552          13265*  13339   13346*  13397#
UIPAR0= 177640           9210#   9418    9560    9635    9702    9757    10041   10712*  10729*  10881*  11379   11435
UIPAR1= 177642           9211#  10882*
UIPAR2= 177644           9212#  10883*
UIPAR3= 177646           9213#  10162   10260*  10799*  10884*
UIPAR4= 177650           9214#  10058*  10079*  10094*  10261*  10800*  10889*  11005*  11153*
UIPAR5= 177652           9215#  10059*  10080*  10095*
UIPAR6= 177654           9216#
UIPAR7= 177656           9217#   9558    9633    10885*
UIPDR0= 177600           9199#   9430    9596    9671    9695    9745    10048   10166   10169   10182   10878   11147   11375
                        11422
UIPDR1= 177602           9200#
UIPDR2= 177604           9201#
UIPDR3= 177606           9202#  10263*  10801*  10851*
```

```
UIPDR4= 177610          9203#  10056*  10081*  10270*  10329*  10802*  11003*  11160*
UIPDR5= 177612          9204#  10057*  10082*
UIPDR6= 177614          9205#
UIPDR7= 177616          9206#   9594    9669
UNIQUE  126714         22664   23231   23255#
UPAT00  057502         13266   13276   13294   13311   13326   13342   13354   13372#
UPAT01  057504         13373#
UPAT02  057506         13374#
UPAT03  057510         13375#
UPAT10  057512         13282   13377#
UPAT11  057514         13378#
UPAT12  057516         13379#
UPAT13  057520         13380#
UPAT20  057522         13300   13382#
UPAT21  057524         13383#
UPAT22  057526         13384#
UPAT23  057530         13385#
UPAT30  057532         13316   13387#
UPAT31  057534         13388#
UPAT32  C57536         13389#
UPAT33  057540         13390#
UPAT40  057542         13331   13356   13392#
UPAT41  057544         13393#
UPAT42  057546         13394#
UPAT43  057550         13395#
UROM1   057560         13278*  13296*  13313*  13328*  13400#
UROM2   057562         13279*  13297*  13314*  13329*  13401#
UROM3   057564         13280*  13298*  13315*  13330*  13402#
USESTK= 000600           539#   5551    5558    5570    5915    5918    5927    9354   10716   10731   10819   10902   11021
                       11163   11270   11288
USP1    017432          5501#   5502
USP1A   017450          5503    5506#
USP2    017506          5523    5526#
USP3    017524          5528    5530#
USP4    017552          5534    5536#
USRM  = 140000           510#   5498    5504    5506    5520    5524    5526    5531   5533    5537    5550    5555    5557
                        5569    5573    5575
UTMP1   057554         13271*  13278   13328   13347*  13398#
UTMP2   057556         13272*  13296   13313   13348*  13399#
UUBDON  107240         19541   19551#
UUBTP1  107154         19516   19520   19521*  19527#  19531
UUBTP2  107226         19515   19532   19544#
UUB10   107236         19535   19538   19540   19549#
UUB2    107152         19526#
UUCBF0  115724         21243   21247   21254#
UUCDON  115732         21248   21256#
UUCTP1  115712         21241   21251#
UUC2    115700         21245#
UUUA1   102140         18056*  18080#
UUUA2   102142         18062   18064   18081#
UUUBF0  102126         18051   18056   18066   18075#  18080
UUUDON  102156         18072   18089#
UUUTP1  102144         18059   18067   18082#
UUU10   102154         18061   18065   18070   18087#
UUU2    102072         18063#
UUU3    102116         18069#  18071
```

```
U0      057014          13268#  13269
U1      057036          13273#  13349
U10     057326          13332#
U11     057330          13333#
U12     057346          13337   13339#
U13     057364          13344#  13345
U14     057416          13340   13351#
U15     057444          13357#
U16     057446          13358#  13363
U2      057106          13283#
U3      057124          13288   13291#
U4      057166          13302#
U5      057204          13307   13309#
U6      057246          13318#
U7      057264          13322   13324#
VDEC    022244           6212    6215#
VDEC10  022420           6253    6256#
VDEC11  022454           6263    6266#
VDEC12  022460           6264    6267#
VDEC13  022520           6276    6279#
VDEC14  022524           6277    6280#
VDEC2   022240           6211    6214#
VDEC3   022300           6222    6225#
VDEC4   022274           6221    6224#
VDEC5   022334           6233    6236#
VDEC6   022330           6232    6235#
VDEC7   022370           6244    6247#
VDEC8   022364           6243    6246#
VDEC9   022424           6254    6257#
VVBDON  104034          18663   18667#
VVB10   104032          18660   18662   18664#
VVB2    104006          18654#
VVCBF0  116002          21272   21276   21283#
VVCDON  116010          21277   21285#
VVCTP1  115770          21270   21280#
VVC2    115756          21274#
VVVBF0  102272          18103   18111   18115   18117   18124#
VVVDON  102314          18123   18135#
VVVTP1  102302          18109   18118   18128#
VVV10   102312          18108   18116   18121   18133#
VVV2    102234          18113#
WAITER  127602          23388#
WAITIO  127372          23343#  23344
WASR6   037224           9254#
WASSR0  037232           9257#  10286*  10288   10344*  10346   10431*  10434   10525*  10528   10659*  10661   10676*  10677
                        10772*  10775   10813*  10832
WASSR2  037234           9258#   9451*   9453    9471*   9472   10287*  10295   10345*  10353   10432*  10441   10526*  10535
                        10607*  10608   10634*  10636   10642*  10644   10660*  10664   10680*  10681   10691*  10693   10773*
                        10782   10814*  10835
WATE    024320           6625    6637#
WATE1   024252           6627#   6628
WATE2   024266           6630#   6631
WATE3   024314           6635#   6640
WBIT  = 000100           9245#  10125   10133   10144   10175   10183   10194
WDAP00  060126          13496#
WDAT01  060130          13497#
```

WDAT02	060132	13498#										
WDAT03	060134	13499#										
WDONE	060136	13488	13501#									
WDONE1	121546	22184#	22187									
WDONE2	125500	22981#	22984									
WPAT00	060116	13412	13414	13418	13420	13431	13435	13439	13441	13452	13454	13458	13460	13471
		13475	13479	13481	13491#							
WPAT01	060120	13492#										
WPAT02	060122	13493#										
WPAT03	060124	13494#										
WWBDON	110474	19837	19921#									
WWB1	107560	19667#										
WWB10	110172	19800#										
WWB11	110240	19819#										
WWB2	107626	19686#										
WWB3	107674	19705#										
WWB4	107742	19724#										
WWB5	110010	19743#										
WWB6	110056	19762#										
WWB7	110124	19781#										
WWCDON	117110	21471	21544#									
WWC1	116014	21298#										
WWC10	116320	21369#										
WWC11	116354	21380#										
WWC12	116410	21390#										
WWC13	116444	21400#										
WWC14	116500	21410#										
WWC15	116534	21420#										
WWC16	116570	21430#										
WWC17	116624	21440#										
WWC2	116050	21308#										
WWC20	116660	21451#										
WWC21	116714	21462#										
WWC3	116104	21318#										
WWC4	116140	21328#										
WWC5	116174	21339#										
WWC6	116230	21349#										
WWC7	116264	21359#										
WWWBF0	102436	18149	18159	18164	18171#							
WWWBF1	102456	18157	18159*	18162	18179#							
WWWDON	102470	18170	18186#									
WWWTP1	102446	18155	18165	18175#								
WWW10	102466	18154	18163	18168	18184#							
WWW2	102400	18160#										
W10	057730	13444	13446#									
W11	057742	13448	13450#									
W12	057762	13455#										
W13	060006	13462#	13465									
W14	060014	13463	13465#									
W15	060026	13467	13469#									
W16	060052	13476#										
W17	060076	13483#	13486									
W2	057606	13415#										
W20	060104	13484	13486#									
W3	057632	13422#	13425									
W4	057640	13423	13425#									

```
W5      057652          13427   13429#
W6      057676          13436#
W7      057722          13443#  13446
X       120122          21824#
XAPT11  060514          13604#
XBUF    100434          17559   17561   17573   17575   17588   17590   17636   17638   17651#
XDAT00  060472          13521   13541   13561   13581   13593#
XDAT01  060474          13594#
XDAT02  060476          13595#
XDAT03  060500          13596#
XDONE   060532          13591   13612#
XMIT1   127406          23322   23348#
XMIT2   127470          23326   23365#
XMTCT1  127760          23317*  23348*  23351   23394   23399   23423#
XMTCT2  127762          23318*  23365*  23368   23405   23410   23424#
XOR1    016106           5101    5102    5103    5104    5107#
XOR2    016110           5106    5109#
XOR3    016136           5113    5114    5115    5116    5119#
XPAT0   100534          17596*  17597   17603*  17604   17667#
XPAT0   100444          17529   17531   17539   17541   17552   17581   17610   17653#
XPAT00  060502          13515   13523   13555   13563   13598#
XPAT01  060504          13599#
XPAT02  060506          13600#
XPAT03  060510          13601#
XPAT1   100454          17655#
XPAT10  060512          13517   13537   13557   13577   13603#
XPAT12  060516          13605#
XPAT13  060520          13606#
XPAT2   100464          17635   17657#
XPAT20  060522          13535   13543   13575   13583   13607#
XPAT21  060524          13608#
XPAT22  060526          13609#
XPAT23  060530          13610#
XPAT3   100474          17553   17567   17582   17659#
XPAT4   100504          17595   17602   17611   17618   17619   17626   17627   17634   17661#
XPAT5   100514          17643   17663#
XPAT6   100524          17644   17665#
XTDONE  100544          17647   17670#
XT1     077652          17517#
XT1A    077666          17517   17520#
XT10    100276          17614   17617#
XT11    100324          17622   17625#
XT12    100352          17630   17633#
XT13    100406          17639   17642#
XT2     077670          17519   17523#
XT2A    077710          17527   17529#
XT2B    077726          17532   17534#
XT3     077740          17536   17538#
XT3A    077762          17542   17544#
XT4     077772          17546   17549#
XT4A    100022          17550   17555#
XT4B    100034          17557   17559#
XT5     100052          17562   17565#
XT5A    100072          17565   17569#
XT5B    100104          17571   17573#
XT6     100122          17576   17579#
```

```
XT6A    100146          17579   17584#
XT6B    100160          17586   17588#
XT7     100176          17591   17594#
XT8     100224          17598   17601#
XT9     100252          17605   17609#
XXBDON  107402          19587   19602#
XXBTP1  107360          19561   19566   19567*  19577   19583   19590#
XXBTP2  107370          19562   19578   19594#
XXB10   107400          19576   19581   19584   19586   19600#
XXB2    107306          19572#
XXCDON  117150          21568#
XXXDON  103122          18292   18366#
XXX1    102474          18199#
XXX2    102540          18218#
XXX3    102604          18237#
XXX4    102650          18256#
XXX5    102714          18275#
X10     060302          13546   13548#
X11     060316          13551   13553#
X12     060336          13558#
X13     060362          13565#  13568
X14     060370          13566   13568#
X15     060404          13571   13573#
X16     060424          13578#
X17     060450          13585#  13588
X2      060162          13518#
X20     060456          13586   13588#
X3      060206          13525#  13528
X4      060214          13526   13528#
X5      060230          13531   13533#
X6      060250          13538#
X7      060274          13545#  13548
Y       120124          21801   21825#
YBR     001534            718*    721*    731#
YDAT00  060722          13635   13660#
YDAT01  060724          13661#
YDAT02  060726          13662#
YDAT03  060730          13663#
YDONE   060762          13653   13680#
YFLAG   060712          13623*  13646   13648*  13655#
YNTAB   021160            705    5954#
YPAT00  060732          13624   13650   13665#
YPAT01  060734          13666#
YPAT02  060736          13667#
YPAT03  060740          13668#
YPAT10  060742          13625   13649   13670#
YPAT11  060744          13671#
YPAT12  060746          13672#
YPAT13  060750          13673#
YPAT20  060752          13629   13675#
YPAT21  060754          13676#
YPAT22  060756          13677#
YPAT23  060760          13678#
YTMP1   060714          13624*  13631   13649*  13656#
YTMP2   060716          13625*  13638   13650*  13657#
YTMP3   060720          13626*  13643   13651*  13658#
```

```
YYBDON  107554          19641   19656#
YYBTP1  107530          19614   19620   19621*  19631   19644#  19652
YYBTP2  107540          19615   19632   19648#
YYBTP3  107550          19619   19637   19652#
YYB10   107552          19630   19635   19638   19640   19653#
YYB2    107456          19626#
YYCDON  117506          21638   21700#
YYC1    117154          21582#
YYC2    117202          21591#
YYC3    117230          21600#
YYC4    117256          21610#
YYC5    117304          21620#
YYC6    117332          21630#
YYYDON  103554          18472   18542#
YYY1    103126          18379#
YYY2    103172          18398#
YYY3    103236          18417#
YYY4    103302          18436#
YYY5    103346          18455#
Y1      060564          13627#  13652
Y2      060604          13632#
Y3      060630          13639#  13642
Y4      060650          13644   13646#
Y5      060710          13647   13653#
Z       120126          21802*  21820*  21821   21826#
ZDAT00  061134          13701   13724#
ZDAT01  061136          13725#
ZDAT02  061140          13726#
ZDAT03  061142          13727#
ZDONE   061174          13718   13744#
ZFLAG   061126          13690*  13712   13714*  13720#
ZPAT00  061144          13691   13729#
ZPAT01  061146          13730#
ZPAT02  061150          13731#
ZPAT03  061152          13732#
ZPAT10  061154          13715   13734#
ZPAT11  061156          13735#
ZPAT12  061160          13736#
ZPAT13  061162          13737#
ZPAT20  061164          13695   13739#
ZPAT21  061166          13740#
ZPAT22  061170          13741#
ZPAT23  061172          13742#
ZTMP1   061130          13691*  13697   13704   13715*  13721#
ZTMP2   061132          13692*  13709   13716*  13722#
ZZCBF   117602          21717   21723   21736#
ZZCDON  117614          21730   21740#
ZZC10   117576          21725   21727   21729   21731#
ZZC2    117520          21715#  21726
ZZC3    117540          21720#
ZZZDON  103614          18563   18568#
ZZZ10   103612          18560   18562   18565#
ZZZ2    103566          18555#
Z1      061006          13693#  13717
Z2      061026          13698#
Z3      061052          13705#  13708
```

```
Z4      061060          13706   13708#
Z5      061072          13710   13712#
Z6      061124          13713   13718#
$APTHD  001030            590     596#
$BDADR  126644          23233#
$BDDAT  126646          23234#
$CPUOP  001026            572#
$DEVCT  001010            563#
$DOAGN  131124          23441   23460#
$ENDAD  131056            549   23450#
$ENV    001020            568#     668   5477   6013   6578   6594   6617   9715  11458  21926  22439  22456  22528
                        22562   22611  22700  22727  22790  22821  22853  22948  22987  23122  23171  23265  23305
$ENVM   001021            569#   23482
$EOPCT  131012          23440#  23444*
$ETABL  001020            567#
$ETEND  001030            579#     602
$FATAL  001002            560#     666*    814*  11471*  21839*  22396*  22647*  23239*  23429*
$GDADR  126650          23235#
$GDDAT  126652          23236#
$HIBTS  C01030            597#
$MAIL   001000            558#     598     602
$MBADR  001032            598#
$MSGAD  001014            565#
$MSGLG  001016            566#
$MSGTY  001000            559#     667*    815*  11472*  21840*  22397*  22648*  23240*  23430*
$PASS   001006            562#     655*   5479   6580   6596   6619   6928   6958   6986   7014   7043   7073   7366
                         7403    7440   9717  11460  21928  22441  22458  22530  22564  22613  22702  22729  22792
                        22823   22855  22950  22989  23124  23173  23267  23307  23442* 23443*
$PASTM  001036            600#
$SETUP= 000020           6896#
$STUP = 177777           6896#
$SVPC = 000400            547#     552
$SWR  = 000000            500#
$SWREG  001022            570#    6015
$TESTN  001004            561#     649    665*   5625   6903   9277*  11501*  21891*  22404*  22665*  23258*
$TMP0   037240           9260#    9815*  9821   9839*  9845   9863*   9869    9939*   9945    9963*   9969    9987*   9993
                        10011*   10017  10060*  10272*  10302  10331*  10360  10654*  10661  10821*  10827*  10831*  10834*
                        10838*   10839  21787*  21829  23312*  23416  23454*  23459
$TMP1   037242           9261#   10651*  10663*  10666*  10667  10673*  10679*  10683*  10684  21788*  21830  23313*  23417
$TMP2   037244           9262#   10655*  10664  10958*  10959  10983*  10984  11088  11089*  11121*  11123  11221*  11222
                        11246*   11247  17689*  17849*  19939*  20011*  23314*  23418
$TMP3   037246           9263#   23315*  23419
$TMP4   037250           9264#   23316*  23420
$TN   = 000667            500#     700    704#    751    755#    780    784#    817    821#    846    850#    883    887#
                           918     922#    947    951#    996   1000#   1003   1006   1010#   1013   1016   1020#   1023
                          1026    1030#   1032   1049   1053#   1057   1060   1064#   1070   1073   1077#   1084   1088
                          1092#   1096   1099   1103#   1110   1134   1138#   1141   1144   1148#   1150   1153   1157#
                          1159    1162   1166#   1168   1171   1175#   1182   1186   1190#   1198   1201   1205#   1212
                          1216    1220#   1228   1232   1236#   1243   1247   1251#   1259   1263   1267#   1274   1278
                          1282#   1290   1295   1299#   1306   1310   1314#   1322   1326   1330#   1337   1341   1345#
                          1353    1371   1375#   1378   1381   1385#   1387   1390   1394#   1396   1399   1403#   1405
                          1438    1442#   1451   1464   1468#   1481   1491   1495#   1501   1513   1517#   1525   1536
                          1540#   1552   1566   1570#   1584   1598   1602#   1617   1631   1635#   1650   1659   1663#
                          1681    1690   1694#   1717   1719   1723#   1746   1748   1752#   1771   1788   1792#   1804
                          1820    1824#   1842   1859   1863#   1879   1882   1886#   1914   1925   1929#   1942   1961
                          1965#   1981   1993   1997#   2006   2019   2023#   2033   2037   2041#   2055   2057   2061#
```

2080 2082 2086# 2100 2102 2106# 2121 2132 2136# 2144 2157 2161# 2168
2180 2184# 2191 2203 2207# 2215 2226 2230# 2249 2260 2264# 2277 2288
2292# 2318 2328 2332# 2347 2359 2363# 2388 2398 2402# 2414 2424 2428#
2443 2453 2457# 2470 2480 2484# 2497 2506 2510# 2515 2522 2526# 2531
2538 2542# 2553 2564 2568# 2596 2604 2608# 2622 2630 2634# 2655 2662
2666# 2680 2690 2694# 2700 2709 2713# 2719 2729 2733# 2739 2753 2757#
2769 2779 2783# 2790 2800 2804# 2813 2825 2829# 2837 2846 2850# 2861
2871 2875# 2879 2888 2892# 2896 2905 2909# 2913 2916 2920# 2931 2934
2938# 2949 2952 2956# 2971 2974 2978# 3004 3008 3012# 3039 3042 3046#
3062 3065 3069# 3093 3096 3100# 3116 3119 3123# 3138 3141 3145# 3160
3169 3173# 3184 3193 3197# 3213 3221 3225# 3246 3254 3258# 3279 3289
3293# 3306 3317 3321# 3341 3353 3357# 3378 3389 3393# 3414 3424 3428#
3448 3463 3467# 3473 3492 3496# 3502 3520 3524# 3530 3542 3546# 3552
3562 3566# 3581 3594 3598# 3625 3636 3640# 3673 3684 3688# 3709 3721
3725# 3733 3747 3751# 3760 3772 3776# 3781 3792 3796# 3802 3815 3819#
3824 3834 3838# 3842 3853 3857# 3864 3875 3879# 3882 3893 3897# 3904
3917 3921# 3928 3944 3948# 3952 3968 3972# 3976 4014 4018# 4072 4092
4096# 4189 4201 4205# 4213 4229 4233# 4247 4250 4254# 4269 4272 4276#
4290 4293 4297# 4313 4328 4332# 4357 4361 4365# 4397 4411 4415# 4421
4425 4429# 4444 4447 4451# 4466 4480 4484# 4520 4524 4528# 4552 4566
4570# 4597 4601 4605# 4638 4642 4646# 4652 4667 4671# 4704 4708 4712#
4746 4760 4764# 4798 4801 4805# 4838 4841 4845# 4880 4883 4887# 4925
4930 4934# 4981 4985 4989# 5005 5010 5014# 5043 5054 5058# 5081 5092
5096# 5118 5128 5132# 5147 5157 5161# 5188 5201 5205# 5218 5222 5226#
5234 5237 5241# 5250 5253 5257# 5265 5268 5272# 5280 5283 5287# 5295
5298 5302# 5310 5321 5325# 5336 5339 5343# 5357 5360 5364# 5378 5381
5385# 5399 5402 5406# 5420 5423 5427# 5441 5444 5448# 5462 5473 5477#
5480 5485 5494 5498# 5516 5520# 5545 5549# 5563 5567# 5579 5591 5595#
5633 5637# 5652 5656# 5662 5679 5683# 5728 5732# 5743 5746 5750# 5761
5764 5768# 5788 5791 5795# 5816 5826 5830# 5833 5836 5840# 5843 5846
5850# 5854 5857 5861# 5864 5867 5871# 5874 5881 5885# 5889 5896 5900#
5905 5908 5912# 5919 5922 5926# 6017 6021# 6076 6078 6082# 6124 6126
6130# 6158 6162 6166# 6172 6176# 6180 6183 6187# 6206 6210# 6216 6220#
6227 6231# 6238 6242# 6248 6252# 6258 6262# 6269 6271 6275# 6282 6286#
6303 6307# 6315 6319# 6327 6329 6333# 6340 6344 6348# 6356 6360 6364#
6381 6385# 6395 6398 6402# 6418 6422# 6444 6448# 6510 6514# 6529 6533#
6550 6554# 6574 6578# 6581 6587 6590 6594# 6597 6613 6617# 6646 6650#
6659 6665 6669# 6675 6683 6687# 6692 6703 6707# 6736 6740# 6749 6753#
9281 9285# 9300 9304# 9319 9323# 9341 9347 9351# 9357 9377 9381# 9389
9393# 9401 9405# 9413 9417# 9425 9429# 9437 9441# 9483 9487# 9500 9504#
9522 9529 9533# 9559 9563 9567# 9595 9601 9605# 9634 9638 9642# 9670
9674 9678# 9709 9713# 9718 9792 9796# 9918 9927 9931# 10031 10035# 10097
10105 10109# 10155 10159# 10206 10210# 10252 10256# 10274 10317 10321# 10399 10403#
10425 10458 10495 10499# 10519 10552 10590 10594# 10627 10631# 10705 10709# 10736
10740# 10750 10757 10761# 10794 10798# 10861 10865# 10998 11002# 11062 11140 11144#
11260 11264# 11291 11295# 11504 11508# 11540 11544# 11565 11569# 11653 11657# 11710
11714# 11738 11742# 11851 11855# 11955 11959# 12060 12064# 12457 12461# 12655 12659#
12756 12760# 12821 12825# 12883 12887# 12956 12960# 13017 13021# 13083 13087# 13150
13154# 13202 13206# 13261 13265# 13406 13410# 13509 13513# 13619 13623# 13686 13690#
13752 13756# 13852 13856# 14028 14032# 14200 14204# 14392 14396# 14467 14471# 14554
14558# 14727 14731# 14953 14957# 15158 15162# 15402 15406# 15500 15504# 15679 15683#
15834 15838# 16012 16016# 16133 16137# 16266 16270# 16411 16415# 16549 16553# 16700
16704# 16925 16929# 17191 17195# 17351 17355# 17513 17517# 17680 17684# 17701 17727
17731# 17787 17791# 17871 17875# 17933 17937# 17993 17997# 18046 18050# 18096 18100#
18142 18146# 18193 18197# 18373 18377# 18548 18552# 18575 18579# 18622 18626# 18647
18651# 18674 18678# 18730 18734# 18781 18785# 18828 18832# 18879 18883# 18930 18934#

```
                    18982  18986#  19028  19032#  19079  19083#  19122  19126#  19171  19175#  19219  19223#  19266
                    19270# 19314   19318# 19364   19368# 19415   19419# 19464   19468# 19511   19515# 19557   19561#
                    19609  19613#  19662  19666#  19930  19934#  19966  19970#  20003  20007#  20034  20038#  20067
                    20071# 20099   20103# 20131   20135# 20168   20172# 20197   20201# 20236   20240# 20480   20484#
                    20653  20657#  20956  20960#  20995  20999#  21035  21039#  21073  21077#  21113  21117#  21153
                    21157# 21194   21198# 21235   21239# 21263   21267# 21292   21296# 21552   21556# 21576   21580#
                    21707  21711#  21748  21752#  21778  21782#  21895  21899#  21909  21913#  21922  21926#  21949
                    21953# 21960   21965  21969#  21978  21982#  21995  21999#  22013  22017   22021# 22045   22049#
                    22074  22078#  22098  22102#  22123  22127#  22151  22155#  22177  22181#  22200  22204#  22211
                    22216  22220#  22244  22248#  22273  22277#  22304  22308#  22328  22332#  22351  22355#  22375
                    22379# 22408   22412# 22420   22424# 22451   22455# 22459   22480  22484#  22524  22528#  22531
                    22558  22562#  22565  22584   22588# 22607   22611# 22669   22673# 22683   22687# 22696   22700#
                    22703  22723   22727# 22730   22739  22744   22748# 22757   22761# 22774   22778# 22793   22798
                    22802  22806#  22834  22838#  22867  22871#  22891  22895#  22916  22920#  22944  22948#  22951
                    22975  22979#  23002  23006#  23011  23016   23020# 23041   23045# 23067   23071# 23096   23100#
                    23118  23122#  23125  23145   23149# 23167   23171# 23174   23195  23199#  23212  23216#  23261
                    23265# 23268   23301  23305#

$TPB    025340       6893#
$TPS    025342       6894#
$TSTM   C01034        599#
$UNIT   001012        564#
$UNITM  001040        601#
$USWR   001024        571#  23444
$X    = 127072        704#   755#   784#   821#   850#   887#   922#   951#  1000#  1010#  1020#  1030#  1053#
                     1064#  1077#  1092#  1103#  1138#  1148#  1157#  1166#  1175#  1190#  1205#  1220#  1236#
                     1251#  1267#  1282#  1299#  1314#  1330#  1345#  1375#  1385#  1394#  1403#  1442#  1468#
                     1495#  1517#  1540#  1570#  1602#  1635#  1663#  1694#  1723#  1752#  1792#  1824#  1863#
                     1886#  1929#  1965#  1997#  2023#  2041#  2061#  2086#  2106#  2136#  2161#  2184#  2207#
                     2230#  2264#  2292#  2332#  2363#  2402#  2428#  2457#  2484#  2510#  2526#  2542#  2568#
                     2608#  2634#  2666#  2694#  2713#  2733#  2757#  2783#  2804#  2829#  2850#  2875#  2892#
                     2909#  2920#  2938#  2956#  2978#  3012#  3046#  3069#  3100#  3123#  3145#  3173#  3197#
                     3225#  3258#  3293#  3321#  3357#  3393#  3428#  3467#  3496#  3524#  3546#  3566#  3598#
                     3640#  3688#  3725#  3751#  3776#  3796#  3819#  3838#  3857#  3879#  3897#  3921#  3948#
                     3972#  4018#  4096#  4205#  4233#  4254#  4276#  4297#  4332#  4365#  4415#  4429#  4451#
                     4484#  4528#  4570#  4605#  4646#  4671#  4712#  4764#  4805#  4845#  4887#  4934#  4989#
                     5014#  5058#  5096#  5132#  5161#  5205#  5226#  5241#  5257#  5272#  5287#  5302#  5325#
                     5343#  5364#  5385#  5406#  5427#  5448#  5477#  5498#  5520#  5549#  5567#  5595#  5637#
                     5656#  5683#  5732#  5750#  5768#  5795#  5830#  5840#  5850#  5861#  5871#  5885#  5900#
                     5912#  5926#  6021#  6082#  6130#  6166#  6176#  6187#  6210#  6220#  6231#  6242#  6252#
                     6262#  6275#  6286#  6307#  6319#  6333#  6348#  6364#  6385#  6402#  6422#  6448#  6514#
                     6533#  6554#  6578#  6594#  6617#  6650#  6669#  6687#  6707#  6740#  6753#  9285#  9304#
                     9323#  9351#  9381#  9393#  9405#  9417#  9429#  9441#  9487#  9504#  9533#  9567#  9605#
                     9642#  9678#  9713#  9796#  9931#  10035# 10109# 10159# 10210# 10256# 10321# 10403# 10499#
                    10594# 10631# 10709# 10740# 10761# 10798# 10865# 11002# 11144# 11264# 11295# 11508# 11544#
                    11569# 11657# 11714# 11742# 11855# 11959# 12064# 12461# 12659# 12760# 12825# 12887# 12960#
                    13021# 13087# 13154# 13206# 13265# 13410# 13513# 13623# 13690# 13756# 13856# 14032# 14204#
                    14396# 14471# 14558# 14731# 14957# 15162# 15406# 15504# 15683# 15838# 16016# 16137# 16270#
                    16415# 16553# 16704# 16929# 17195# 17355# 17517# 17684# 17731# 17791# 17875# 17937# 17997#
                    18050# 18100# 18146# 18197# 18377# 18552# 18579# 18626# 18651# 18678# 18734# 18785# 18832#
                    18883# 18934# 18986# 19032# 19083# 19126# 19175# 19223# 19270# 19318# 19368# 19419# 19468#
                    19515# 19561# 19613# 19666# 19934# 19970# 20007# 20038# 20071# 20103# 20135# 20172# 20201#
                    20240# 20484# 20657# 20960# 20999# 21039# 21077# 21117# 21157# 21198# 21239# 21267# 21296#
                    21556# 21580# 21711# 21752# 21782# 21899# 21913# 21926# 21953# 21969# 21982# 21999# 22021#
                    22049# 22078# 22102# 22127# 22155# 22181# 22204# 22220# 22248# 22277# 22308# 22332# 22355#
                    22379# 22412# 22424# 22455# 22484# 22528# 22562# 22588# 22611# 22673# 22687# 22700# 22727#
                    22748# 22761# 22778# 22806# 22838# 22871# 22895# 22920# 22948# 22979# 23006# 23020# 23045#
```

. = 131275

23071# 23100# 23122# 23149# 23171# 23199# 23216# 23265# 23305#
543# 547 548# 550# 552# 553# 586 587# 589# 591# 606# 615# 620#
623# 629# 637# 643# 646# 653# 704 715 738 743 755 784 816
821 850 887 922 951 1000 1010 1020 1030 1053 1064 1077 1092
1103 1138 1148 1157 1166 1175 1190 1205 1220 1236 1251 1267 1282
1299 1314 1330 1345 1375 1385 1394 1403 1442 1468 1495 1517 1540
1570 1602 1635 1663 1694 1723 1752 1792 1824 1863 1886 1929 1965
1997 2023 2041 2061 2086 2106 2136 2161 2184 2207 2230 2264 2292
2332 2363 2402 2428 2457 2484 2510 2526 2542 2568 2608 2634 2666
2694 2713 2733 2757 2783 2804 2829 2850 2875 2892 2909 2920 2938
2956 2978 3012 3046 3069 3100 3123 3145 3173 3197 3225 3258 3293
3321 3357 3393 3428 3467 3496 3524 3546 3566 3598 3640 3688 3725
3751 3776 3796 3819 3838 3857 3879 3897 3921 3948 3972 4018 4021
4096 4205 4233 4254 4276 4297 4332 4365 4415 4429 4451 4484 4528
4570 4605 4646 4671 4712 4764 4805 4845 4887 4934 4989 5014 5058
5096 5132 5161 5205 5226 5241 5257 5272 5287 5302 5325 5343 5364
5385 5406 5427 5448 5477 5498 5520 5549 5567 5595 5637 5656 5683
5732 5750 5768 5795 5830 5840 5850 5861 5871 5885 5900 5912 5926
5937 5938 5939 5940 5941 5942 5943 5944 5945 5946 5947 5948 5949
5950 5951 6021 6082 6130 6166 6176 6187 6210 6220 6231 6242 6252
6262 6275 6286 6307 6310 6319 6322 6333 6336 6339 6348 6364 6385
6402 6409 6422 6448 6514 6533 6554 6578 6594 6617 6650 6669 6687
6707 6740 6753 6855 6869# 6871# 6873# 6875# 6877# 6879# 9285 9304 9323
9351 9381 9393 9405 9417 9429 9441 9487 9504 9533 9567 9605 9642
9678 9713 9796 9931 10035 10109 10159 10210 10256 10321 10403 10499 10594
10631 10709 10740 10761 10798 10865 11002 11144 11264 11295 11473 11508 11544
11569 11657 11714 11742 11855 11959 12064 12461 12659 12760 12825 12887 12960
12971# 13021 13087 13154 13206 13265 13410 13513 13623 13690 13756 13856 14032
14204 14396 14471 14558 14731 14957 15162 15406 15504 15683 15838 16016 16137
16270 16415 16553 16704 16929 17195 17355 17517 17684 17731 17791 17875 17937
17997 18050 18100 18146 18197 18377 18552 18579 18626 18651 18678 18734 18785
18832 18883 18934 18986 19032 19083 19126 19175 19223 19270 19318 19368 19419
19468 19515 19561 19613 19666 19934 19970 20007 20038 20071 20103 20135 20172
20201 20240 20484 20657 20960 20999 21039 21077 21117 21157 21198 21239 21267
21296 21556 21580 21711 21752 21782 21841 21899 21913 21926 21953 21969 21982
21999 22021 22049 22078 22102 22127 22155 22181 22204 22220 22248 22277 22308
22332 22355 22379 22398 22412 22424 22455 22484 22528 22562 22588 22611 22649
22673 22687 22700 22727 22748 22761 22778 22806 22838 22871 22895 22920 22948
22979 23006 23020 23045 23071 23100 23122 23149 23171 23199 23216 23241 23265
23305 23431 23433# 23434#

.RSET 120200

11533 11559 11646 11704 11731 11844 11948 12052 12450 12648 12750 12814 12876
12949 13010 13076 13143 13195 13255 13502 13613 13681 13745 13847 14022 14195
14387 14462 14539 14720 14946 15149 15393 15492 15671 15826 16004 16125 16257
16402 16540 16690 16916 17182 17342 17506 17671 17718 17778 17865 17927 17987
18040 18090 18136 18187 18367 18543 18569 18616 18641 18668 18724 18775 18823
18874 18925 18977 19023 19074 19117 19166 19214 19261 19309 19358 19409 19459
19506 19552 19603 19657 19922 19959 19996 20029 20062 20094 20126 20160 20190
20228 20473 20646 20948 20988 21027 21065 21105 21146 21187 21229 21257 21286
21545 21569 21701 21741 21770 21831 21847#

.$X = 001030

586# 591

ACCMAC 11483# 12065 12091 12117 12143 12169 12195 12221 12247 12273 12299 12325 12351
ADDTST 9252# 9810 9834 9858 9934 9958 9982 10006
APTSKP 500# 5477 6578 6594 9715 22455 22528 22562 22700 22727 22789 22948 23122 23171 23265
COMMEN 1#
ENDCOM 1#
ERROR 500# 732 760 775 789 804 826 841 855 870 892 907 927 942 956
971 1003 1013 1023 1032 1057 1067 1070 1084 1096 1110 1141 1150 1159 1168
1182 1198 1212 1228 1243 1259 1274 1290 1306 1322 1337 1353 1378 1387 1396
1405 1443 1448 1451 1470 1481 1496 1501 1519 1525 1544 1552 1575 1584 1608
1617 1641 1650 1670 1681 1700 1705 1713 1717 1730 1735 1742 1746 1759 1768
1771 1797 1804 1831 1842 1869 1879 1897 1901 1906 1909 1914 1933 1942 1974
1981 2001 2006 2028 2033 2048 2052 2055 2072 2076 2080 2096 2100 2114 2121
2138 2144 2168 2191 2215 2238 2249 2272 2277 2300 2304 2313 2318 2342 2347
2374 2383 2388 2410 2414 2438 2443 2466 2470 2493 2497 2515 2531 2549 2553
2570 2577 2582 2591 2596 2617 2622 2638 2641 2645 2651 2655 2674 2680 2700
2719 2739 2765 2769 2790 2809 2813 2834 2837 2857 2861 2879 2896 2913 2927
2931 2945 2949 2962 2967 2971 2986 2990 2996 3001 3004 3023 3027 3034 3039
3055 3059 3062 3078 3082 3086 3090 3093 3109 3113 3116 3131 3135 3138 3153
3157 3160 3180 3184 3204 3209 3213 3230 3234 3239 3243 3246 3264 3268 3271
3275 3279 3299 3303 3306 3326 3329 3334 3338 3341 3363 3367 3370 3375 3378
3399 3403 3406 3411 3414 3434 3438 3441 3445 3448 3473 3502 3530 3552 3572
3581 3605 3614 3625 3649 3660 3673 3693 3701 3709 3733 3760 3781 3802 3821
3824 3842 3861 3864 3882 3901 3904 3925 3928 3952 3976 4022 4025 4033 4039
4042 4049 4056 4064 4072 4107 4119 4132 4144 4156 4166 4176 4189 4211 4213
4238 4247 4260 4269 4282 4290 4304 4313 4338 4348 4357 4371 4379 4387 4397
4421 4435 4444 4457 4466 4489 4497 4505 4511 4520 4534 4543 4552 4577 4587
4597 4611 4620 4630 4638 4652 4677 4686 4695 4704 4719 4728 4737 4746 4771
4780 4788 4798 4812 4821 4830 4838 4852 4861 4869 4880 4894 4904 4913 4925
4939 4945 4951 4955 4967 4971 4977 4981 4995 4999 5005 5019 5025 5030 5039
5043 5067 5081 5106 5118 5137 5147 5168 5175 5184 5188 5209 5218 5231 5234
5247 5250 5262 5265 5277 5280 5292 5295 5307 5310 5328 5336 5350 5354 5357
5371 5375 5378 5392 5396 5399 5413 5417 5420 5434 5438 5441 5455 5459 5462
5485 5505 5525 5528 5535 5556 5560 5576 5579 5622 5643 5662 5693 5709 5737
5743 5755 5761 5775 5779 5784 5788 5802 5807 5812 5816 5833 5843 5854 5864
5874 5889 5905 5919 5934 5978 5980 5982 5984 5986 5988 5990 5992 5994 6024
6030 6036 6043 6050 6057 6063 6070 6076 6088 6097 6106 6115 6124 6131 6135
6139 6143 6146 6150 6154 6158 6169 6180 6192 6198 6202 6269 6298 6313 6325
6327 6340 6356 6374 6378 6392 6395 6412 6414 6429 6458 6463 6466 6484 6491
6494 6523 6543 6545 6565 6587 6606 6636 6638 6641 6657 6659 6674 6690 6692
6713 6714 6717 6720 6723 6803 6806 6808 6937 6941 6966 6970 6994 6998 7022
7026 7051 7055 7081 7085 7165 7180 7183 7197 7200 7214 7217 7231 7234 7248
7251 7265 7268 7282 7285 7299 7302 7316 7319 7333 7336 7374 7378 7383 7387
7411 7415 7420 7423 7448 7452 7457 7460 7569 7585 7588 7605 7608 7626 7629
7647 7650 7667 7670 7688 7691 7709 7712 7729 7732 7735 7757 7760 7763 7780
7783 7786 7803 7806 7809 7826 7829 7832 7849 7852 7855 7872 7875 7878 7895
7898 7901 7918 7921 7924 7941 7944 7947 7964 7967 7970 7999 8002 8005 8020
8023 8026 8041 8044 8047 8063 8066 8069 8085 8088 8091 8106 8109 8112 8128
8131 8134 8150 8153 8156 8171 8174 8177 8192 8195 8198 8214 8217 8220 8236
8239 8242 8257 8260 8263 8278 8281 8284 8300 8303 8306 8322 8325 8328 8343
8346 8349 8365 8368 8371 8391 8394 8397 8412 8415 8418 8433 8436 8439 8454
8457 8460 8475 8478 8481 8496 8499 8502 8517 8520 8523 8538 8541 8544 8559
8562 8565 8580 8583 8586 8608 8611 8614 8629 8632 8635 8651 8654 8657 8673
8676 8679 8694 8697 8700 8716 8719 8722 8738 8741 8744 8759 8762 8765 8780
8783 8786 8802 8805 8808 8824 8827 8830 8845 8848 8851 8867 8870 8873 8889
8892 8895 8910 8913 8916 8933 8950 8969 8972 8975 8990 8993 8996 9011 9014
9017 9032 9035 9038 9053 9056 9059 9074 9077 9080 9095 9098 9101 9116 9119

```
                9122    9137    9140    9143    9158    9161    9164    9177    9180    9291    9310    9330    9341    9357    9446
                9454    9465    9474    9492    9506    9513    9521    9538    9548    9572    9584    9614    9626    9651    9662
                9726    9738    9749    9762    9824    9848    9872    9897    9918    9948    9972    9996   10020   10066   10099
               10126   10136   10145   10176   10186   10195   10218   10230   10274   10281   10289   10296   10333   10339   10347
               10354   10425   10435   10442   10451   10519   10529   10536   10545   10609   10637   10645   10668   10685   10694
               10723   10750   10767   10776   10783   10840   10904   10917   10928   10938   10949   10963   10975   10989   11015
               11032   11048   11062   11064   11079   11096   11112   11130   11168   11181   11192   11202   11213   11226   11238
               11251   11272   11283   11303   11310   11401   11415   11428   11442   11526   11555   11577   11587   11597   11606
               11616   11625   11634   11643   11671   11685   11690   11702   11727   11763   11766   11772   11778   11796   11799
               11805   11811   11881   11908   11922   11984   12011   12025   12412   12599   12683   12686   12709   12712   12716
               12732   12777   12783   12788   12840   12846   12851   12914   12923   12977   12985   12989   13041   13047   13108
               13115   13172   13176   13225   13230   13288   13307   13322   13337   13360   13364   13370   13423   13427   13444
               13448   13463   13467   13484   13488   13526   13531   13546   13551   13566   13571   13586   13591   13640   13644
               13706   13710   13769   13788   13810   13869   13873   13888   13892   13908   13912   13928   13932   13947   13951
               13966   13970   14045   14050   14065   14069   14086   14090   14106   14110   14125   14129   14144   14148   14217
               14222   14237   14242   14257   14262   14277   14281   14296   14300   14315   14320   14335   14340   14409   14414
               14429   14434   14484   14488   14504   14508   14573   14577   14597   14602   14620   14625   14640   14645   14660
               14665   14746   14751   14763   14767   14776   14781   14787   14804   14809   14823   14832   14837   14843   14860
               14865   14877   14881   14890   14895   14901   14965   14974   14979   14992   15003   15009   15021   15036   15054
               15078   15084   15099   15105   15374   15381   15423   15428   15445   15449   15464   15469   15477   15481   15657
               15660   15663   15814   15819   15990   15993   15996   16113   16118   16243   16249   16388   16394   16507   16510
               16524   16529   16532   16658   16662   16675   16679   16682   16889   16892   16896   16899   16902   17159   17167
               17171   17205   17227   17249   17311   17314   17318   17321   17324   17369   17398   17476   17484   17488   17520
               17527   17532   17536   17542   17546   17554   17557   17562   17568   17571   17576   17583   17586   17591   17598
               17605   17614   17622   17630   17639   17647   17696   17701   17711   17714   17751   17775   17812   17834   17852
               17862   17908   17984   18036   18087   18133   18184   18362   18538   18565   18602   18638   18664   18705   18772
               18820   18871   18922   18974   19020   19071   19114   19163   19211   19258   19306   19355   19406   19456   19503
               19549   19600   19653   19914   19955   19993   20026   20059   20091   20123   20157   20185   20213   20217   20465
               20642   20940   20984   21023   21062   21102   21143   21184   21226   21248   21277   21537   21685   21688   21692
               21731   21760   21763   21800   21816   21902   21916   21938   21944   21960   21972   21985   22000   22004   22008
               22013   22025   22030   22034   22040   22053   22058   22062   22068   22084   22091   22110   22117   22138   22145
               22167   22189   22196   22211   22233   22237   22259   22268   22289   22297   22321   22345   22364   22367   22390
               22415   22428   22433   22437   22447   22462   22469   22475   22494   22503   22513   22544   22550   22578   22599
               22640   22676   22690   22712   22718   22739   22751   22764   22779   22783   22787   22798   22810   22815   22819
               22829   22842   22847   22851   22861   22877   22884   22903   22910   22931   22938   22965   22984   22994   23011
               23028   23034   23053   23060   23081   23089   23111   23139   23156   23159   23185   23189   23205   23208   23223
               23227   23290   23361   23378   23395   23401   23406   23412
ESCAPE          1#
GETPRI          1#
GETSWR          1#
HMAC1       11479#  12471   12475   12479   12483   12487
HMAC2       11481#  12571   12575   12579   12583   12587
HMAC3       11482#  12496   12509   12522   12535   12548
JNE           500#    713     736     741
MSG         11504#  11540#  11565#  11653#  11710#  11738#  11851#  11954#  12059#  12457#  12655#  12756#  12820#  12882#  12955#
            13016#  13082#  13149#  13201#  13261#  13406#  13509#  13619#  13686#  13752#  13852#  14028#  14200#  14392#  14467#
            14554#  14726#  14952#  15157#  15401#  15499#  15678#  15833#  16011#  16132#  16265#  16410#  16548#  16699#  16924#
            17190#  17350#  17512#  17679#  17726#  17786#  17871#  17933#  17993#  18046#  18096#  18142#  18193#  18373#  18548#
            18575#  18622#  18646#  19662#  20167#  20196#  20235#  20479#  20652#  21235#  21263#  21292#  21552#  21576#  21707#
MSG1         9281#  18674#  19171#  19929#
MSG10        9413#
MSG11        9425#
MSG12        9437#
MSG13        9483#
MSG14        9500#
MSG15        9525#
```

```
MSG16     9563#
MSG17     9601#
MSG2      9300#  20955#
MSG20     9638#
MSG21     9674#
MSG21A    9709#
MSG22     9792#
MSG23     9927#
MSG24    10031#
MSG25    10105#
MSG26    10155#
MSG27    10206#
MSG3      9319#
MSG30    10252#
MSG31    10317#
MSG32    10398#
MSG33    10494#
MSG34    10589#
MSG35    10626#
MSG36    10704#
MSG36A   10736#
MSG37    10756#
MSG4      9347#
MSG40    10793#
MSG41    10861#
MSG42    10998#
MSG43    11140#
MSG44    11260#
MSG45    11291#
MSG5      9376#
MSG6      9389#
MSG7      9401#
MULT         1#
NEWTST       1#   500#   700   751   780   817   846   883   918   947   996  1006  1016  1026  1049
     1060  1073  1088  1099  1134  1144  1153  1162  1171  1186  1201  1216  1232  1247  1263
     1278  1295  1310  1326  1341  1371  1381  1390  1399  1438  1464  1491  1513  1536  1566
     1598  1631  1659  1690  1719  1748  1788  1820  1859  1882  1925  1961  1993  2019  2037
     2057  2082  2102  2132  2157  2180  2203  2226  2260  2288  2328  2359  2398  2424  2453
     2480  2506  2522  2538  2564  2604  2630  2662  2690  2709  2729  2753  2779  2800  2825
     2846  2871  2888  2905  2916  2934  2952  2974  3008  3042  3065  3096  3119  3141  3169
     3193  3221  3254  3289  3317  3353  3389  3424  3463  3492  3520  3542  3562  3594  3636
     3684  3721  3747  3772  3792  3815  3834  3853  3875  3893  3917  3944  3968  4014  4092
     4201  4229  4250  4272  4293  4328  4361  4411  4425  4447  4480  4524  4566  4601  4642
     4667  4708  4760  4801  4841  4883  4930  4985  5010  5054  5092  5128  5157  5201  5222
     5237  5253  5268  5283  5298  5321  5339  5360  5381  5402  5423  5444  5473  5494  5516
     5545  5563  5591  5633  5652  5679  5727  5746  5764  5791  5826  5836  5846  5857  5867
     5881  5896  5908  5922  6016  6078  6126  6162  6172  6183  6206  6216  6227  6238  6248
     6258  6271  6282  6303  6315  6329  6344  6360  6381  6398  6418  6444  6510  6529  6550
     6574  6590  6613  6646  6665  6683  6703  6736  6749  9281  9300  9319  9347  9377  9389
     9401  9413  9425  9437  9483  9500  9529  9563  9601  9638  9674  9709  9792  9927 10031
    10105 10155 10206 10252 10317 10399 10495 10590 10627 10705 10736 10757 10794 10861 10998
    11140 11260 11291 11504 11540 11565 11653 11710 11738 11851 11955 12060 12457 12655 12756
    12821 12883 12956 13017 13083 13150 13202 13261 13406 13509 13619 13686 13752 13852 14028
    14200 14392 14467 14554 14727 14953 15158 15402 15500 15679 15834 16012 16133 16266 16411
    16549 16700 16925 17191 17351 17513 17680 17727 17787 17871 17933 17993 18046 18096 18142
    18193 18373 18548 18575 18622 18647 18674 18730 18781 18828 18879 18930 18982 19028 19079
```

```
                19122   19171   19219   19266   19314   19364   19415   19464   19511   19557   19609   19662   19930   19966   20003
                20034   20067   20099   20131   20168   20197   20236   20480   20653   20956   20995   21035   21073   21113   21153
                21194   21235   21263   21292   21552   21576   21707   21748   21778   21895   21909   21922   21949   21965   21978
                21995   22017   22045   22074   22098   22123   22151   22177   22200   22216   22244   22273   22304   22328   22351
                22375   22408   22420   22451   22480   22524   22558   22584   22607   22669   22683   22696   22723   22744   22757
                22774   22802   22834   22867   22891   22916   22944   22975   23002   23016   23041   23067   23096   23118   23145
                23167   23195   23212   23261   23301
NXTTST            500#    9559    9595    9634    9670   10097   10458   10552
POP                 1#
PUSH                1#
REPORT              1#
RSET            11476#  11532   11558   11645   11703   11730   11843   11947   12051   12449   12647   12749   12813   12875   12948
                13009   13075   13142   13194   13254   13501   13612   13680   13744   13846   14021   14194   14386   14461   14538
                14719   14945   15148   15392   15491   15670   15825   16003   16124   16256   16401   16539   16689   16915   17181
                17341   17505   17670   17717   17777   17864   17926   17986   18039   18089   18135   18186   18366   18542   18568
                18615   18640   18667   18723   18774   18822   18873   18924   18976   19022   19073   19116   19165   19213   19260
                19308   19357   19408   19458   19505   19551   19602   19656   19921   19958   19995   20028   20061   20093   20125
                20159   20189   20227   20472   20645   20947   20987   21026   21064   21104   21145   21186   21228   21256   21285
                21544   21568   21700   21740   21770   21831
SETPRI              1#
SETUP               1#
SKIP                1#
SLASH               1#
STARS               1#    500#    545    556    583    585    592    603    605    634    636    641    678    700    702
                  751     753     780    782    810    813    817    819    846    848    876    879    883    885    918
                  920     947     949    977    996    998   1006   1008   1016   1018   1026   1028   1035   1049   1051
                 1060    1062    1073   1075   1088   1090   1099   1101   1114   1134   1136   1144   1146   1153   1155
                 1162    1164    1171   1173   1186   1188   1201   1203   1216   1218   1232   1234   1247   1249   1263
                 1265    1278    1280   1295   1297   1310   1312   1326   1328   1341   1343   1357   1371   1373   1381
                 1383    1390    1392   1399   1401   1407   1422   1426   1438   1440   1455   1464   1466   1485   1491
                 1493    1505    1513   1515   1529   1536   1538   1557   1566   1568   1588   1598   1600   1621   1631
                 1633    1654    1659   1661   1685   1690   1692   1719   1721   1748   1750   1774   1788   1790   1808
                 1820    1822    1846   1859   1861   1882   1884   1917   1925   1927   1946   1961   1963   1985   1993
                 1995    2010    2019   2021   2037   2039   2057   2059   2082   2084   2102   2104   2124   2132   2134
                 2149    2157    2159   2172   2180   2182   2195   2203   2205   2219   2226   2228   2253   2260   2262
                 2280    2288    2290   2321   2328   2330   2350   2359   2361   2390   2398   2400   2417   2424   2426
                 2446    2453    2455   2473   2480   2482   2500   2506   2508   2518   2522   2524   2534   2538   2540
                 2556    2564    2566   2599   2604   2606   2624   2630   2632   2657   2662   2664   2683   2690   2692
                 2703    2709    2711   2722   2729   2731   2742   2753   2755   2772   2779   2781   2793   2800   2802
                 2816    2825    2827   2839   2846   2848   2863   2871   2873   2881   2888   2890   2898   2905   2907
                 2916    2918    2934   2936   2952   2954   2974   2976   3008   3010   3042   3044   3065   3067   3096
                 3098    3119    3121   3141   3143   3163   3169   3171   3187   3193   3195   3216   3221   3223   3249
                 3254    3256    3282   3289   3291   3309   3317   3319   3344   3353   3355   3381   3389   3391   3417
                 3424    3426    3451   3463   3465   3483   3492   3494   3510   3520   3522   3532   3542   3544   3554
                 3562    3564    3585   3594   3596   3629   3636   3638   3677   3684   3686   3713   3721   3723   3737
                 3747    3749    3765   3772   3774   3785   3792   3794   3808   3815   3817   3827   3834   3836   3846
                 3853    3855    3868   3875   3877   3886   3893   3895   3908   3917   3919   3935   3944   3946   3958
                 3968    3970    3983   4014   4016   4076   4092   4094   4194   4201   4203   4215   4229   4231   4250
                 4252    4272    4274   4293   4295   4317   4328   4330   4361   4363   4402   4411   4413   4425   4427
                 4447    4449    4470   4480   4482   4524   4526   4556   4566   4568   4601   4603   4642   4644   4657
                 4667    4669    4708   4710   4750   4760   4762   4801   4803   4841   4843   4883   4885   4930   4932
                 4985    4987    5010   5012   5046   5054   5056   5084   5092   5094   5121   5128   5130   5150   5157
                 5159    5192    5201   5203   5222   5224   5237   5239   5253   5255   5268   5270   5283   5285   5298
                 5300    5313    5321   5323   5339   5341   5360   5362   5381   5383   5402   5404   5423   5425   5444
                 5446    5465    5473   5475   5489   5494   5496   5509   5516   5518   5540   5545   5547   5563   5565
                 5584    5591    5593   5627   5633   5635   5646   5652   5654   5665   5679   5681   5717   5728   5730
```

```
                5746  5748  5764  5766  5791  5793  5819  5826  5828  5836  5838  5846  5848  5857  5859
                5867  5869  5876  5880  5881  5883  5894  5896  5898  5908  5910  5922  5924  5972  5976
                5996# 6017  6019  6078  6080  6126  6128  6162  6164  6172  6174  6183  6185  6206  6208
                6216  6218  6227  6229  6238  6240  6248  6250  6258  6260  6271  6273  6282  6284  6303
                6305  6315  6317  6329  6331  6344  6346  6360  6362  6381  6383  6398  6400  6418  6420
                6444  6446  6510  6512  6529  6531  6550  6552  6574  6576  6590  6592  6613  6615  6646
                6648  6665  6667  6683  6685  6703  6705  6736  6738  6749  6751  9281  9283  9300  9302
                9319  9321  9347  9349  9363  9374  9377  9379  9389  9391  9401  9403  9413  9415  9425
                9427  9437  9439  9483  9485  9500  9502  9529  9531  9563  9565  9601  9603  9638  9640
                9674  9676  9709  9711  9792  9794  9927  9929 10031 10033 10105 10107 10155 10157 10206
               10208 10240 10250 10252 10254 10317 10319 10380 10395 10399 10401 10495 10497 10590 10592
               10627 10629 10705 10707 10736 10738 10757 10759 10794 10796 10853 10858 10861 10863 10998
               11000 11140 11142 11260 11262 11291 11293 11321 11328 11340 11346 11356 11364 11385 11393
               11452 11467 11469 11504 11506 11540 11542 11565 11567 11653 11655 11710 11712 11738 11740
               11851 11853 11955 11957 12060 12062 12457 12459 12655 12657 12756 12758 12821 12823 12883
               12885 12956 12958 13017 13019 13083 13085 13150 13152 13202 13204 13261 13263 13406 13408
               13509 13511 13619 13621 13686 13688 13752 13754 13852 13854 14028 14030 14200 14202 14392
               14394 14467 14469 14554 14556 14727 14729 14953 14955 15158 15160 15402 15404 15500 15502
               15679 15681 15834 15836 16012 16014 16133 16135 16266 16268 16411 16413 16549 16551 16700
               16702 16925 16927 17191 17193 17351 17353 17513 17515 17680 17682 17727 17729 17787 17789
               17871 17873 17933 17935 17993 17995 18046 18048 18096 18098 18142 18144 18193 18195 18373
               18375 18548 18550 18575 18577 18622 18624 18647 18649 18674 18676 18730 18732 18781 18783
               18828 18830 18879 18881 18930 18932 18982 18984 19028 19030 19079 19081 19122 19124 19171
               19173 19219 19221 19266 19268 19314 19316 19364 19366 19415 19417 19464 19466 19511 19513
               19557 19559 19609 19611 19662 19664 19930 19932 19966 19968 20003 20005 20034 20036 20067
               20069 20099 20101 20131 20133 20168 20170 20197 20199 20236 20238 20480 20482 20653 20655
               20956 20958 20995 20997 21035 21037 21073 21075 21113 21115 21153 21155 21194 21196 21235
               21237 21263 21265 21292 21294 21552 21554 21576 21578 21707 21709 21748 21750 21778 21780
               21843 21895 21897 21909 21911 21922 21924 21949 21951 21965 21967 21978 21980 21995 21997
               22017 22019 22045 22047 22074 22076 22098 22100 22123 22125 22151 22153 22177 22179 22200
               22202 22216 22218 22244 22246 22273 22275 22304 22306 22328 22330 22351 22353 22375 22377
               22408 22410 22420 22422 22451 22453 22480 22482 22524 22526 22558 22560 22584 22586 22607
               22609 22669 22671 22683 22685 22696 22698 22723 22725 22744 22746 22757 22759 22774 22776
               22802 22804 22834 22836 22867 22869 22891 22893 22916 22918 22944 22946 22975 22977 23002
               23004 23016 23018 23041 23043 23067 23069 23096 23098 23118 23120 23145 23147 23167 23169
               23195 23197 23212 23214 23261 23263 23301 23303 23465 23468 23492 23495

SWRSU             1#
TIMMSG         9252#
TIMTST         9252#  9393  9405  9417  9429
TRAPTS          749#   751   780   817   846   883   918   947
TYPBIN            1#
TYPDEC            1#
TYPNAM            1#
TYPNUM            1#
TYPOCS            1#
TYPOCT            1#
TYPTXT            1#
USER           9252#  9253
VTRP           6205#  6206  6216  6238  6248  6258  6271
WMSG           9252#
WTST           9252#  10109 10160
$ASHC          6901#  7574  7594  7614  7636  7656  7676  7698  7718
$ASHC$         6901#  7746  7769  7792  7815  7838  7861  7884  7907  7930  7953
$ASH$          6901#  7172  7189  7206  7223  7240  7257  7274  7291  7308  7325
$DIV           7983#  8598  8619  8640  8663  8684  8705  8728  8749  8770  8791  8814  8835  8856  8879
                8900  8921  8939  8959  8980  9001  9022  9043  9064  9085  9106  9127  9148
```

$MUL	7983#	7990	8011	8032	8053	8076	8097	8118	8141	8162	8183	8204	8227	8248	8269
	8290	8313	8334	8355	8382	8403	8424	8445	8466	8487	8508	8529	8550	8571	
$SKIP	500#	5480	6581	6597	9718	22459	22531	22565	22703	22730	22793	22951	23125	23174	23268
$$ERCD	500#	733	761	775	790	804	827	841	856	870	893	907	928	942	957
	971	1004	1014	1024	1033	1058	1068	1071	1086	1097	1112	1142	1151	1160	1169
	1184	1200	1214	1230	1245	1261	1276	1292	1308	1324	1339	1355	1379	1388	1397
	1406	1444	1449	1452	1471	1483	1497	1503	1520	1527	1545	1554	1576	1586	1609
	1619	1642	1652	1671	1683	1702	1706	1715	1718	1732	1736	1744	1747	1761	1769
	1772	1798	1806	1832	1844	1870	1881	1899	1902	1907	1910	1915	1934	1944	1975
	1983	2002	2008	2029	2035	2050	2053	2056	2074	2077	2081	2098	2101	2116	2122
	2139	2146	2170	2193	2217	2240	2251	2274	2278	2302	2305	2315	2319	2344	2348
	2376	2385	2389	2412	2415	2440	2444	2468	2471	2495	2498	2516	2532	2551	2554
	2571	2578	2583	2592	2597	2619	2623	2639	2642	2646	2652	2656	2676	2682	2701
	2720	2740	2766	2770	2791	2810	2814	2835	2838	2858	2862	2880	2897	2914	2929
	2932	2947	2950	2964	2969	2972	2988	2991	2998	3002	3005	3025	3028	3036	3040
	3057	3060	3063	3080	3083	3088	3091	3094	3111	3114	3117	3133	3136	3139	3155
	3158	3161	3182	3185	3206	3211	3214	3232	3235	3241	3244	3247	3266	3269	3272
	3276	3280	3301	3304	3307	3327	3330	3336	3339	3342	3365	3368	3371	3376	3379
	3401	3404	3407	3412	3415	3436	3439	3442	3446	3449	3475	3504	3531	3553	3574
	3583	3607	3616	3627	3651	3662	3675	3695	3703	3711	3735	3762	3783	3804	3822
	3825	3843	3862	3865	3883	3902	3905	3926	3930	3954	3978	4023	4026	4034	4040
	4043	4050	4057	4065	4073	4108	4121	4134	4146	4158	4168	4178	4191	4211	4214
	4240	4249	4262	4271	4284	4292	4306	4315	4340	4350	4359	4373	4381	4389	4399
	4423	4437	4446	4459	4468	4491	4499	4507	4513	4522	4536	4545	4554	4579	4589
	4599	4613	4622	4632	4640	4654	4679	4688	4697	4706	4721	4730	4739	4748	4773
	4782	4790	4800	4814	4823	4832	4840	4854	4863	4871	4882	4896	4906	4915	4927
	4940	4947	4953	4956	4969	4972	4979	4982	4997	5000	5007	5020	5027	5032	5041
	5044	5069	5083	5108	5120	5139	5149	5168	5177	5184	5190	5210	5220	5232	5235
	5248	5251	5263	5266	5278	5281	5293	5296	5308	5311	5329	5337	5352	5355	5358
	5373	5376	5379	5394	5397	5400	5415	5418	5421	5436	5439	5442	5457	5460	5463
	5486	5505	5525	5529	5535	5556	5560	5576	5580	5624	5643	5664	5694	5711	5738
	5744	5756	5762	5776	5780	5785	5789	5803	5808	5813	5817	5834	5844	5855	5865
	5875	5890	5906	5920	5935	5979	5981	5983	5985	5987	5989	5991	5993	5995	6025
	6031	6037	6044	6051	6058	6064	6071	6077	6089	6098	6107	6116	6125	6132	6136
	6140	6144	6147	6151	6155	6159	6169	6181	6193	6198	6203	6270	6300	6313	6325
	6328	6341	6357	6374	6379	6392	6396	6413	6415	6431	6459	6464	6467	6485	6492
	6496	6524	6544	6546	6566	6588	6607	6636	6639	6643	6657	6660	6676	6690	6693
	6713	6715	6718	6721	6724	6803	6806	6810	6938	6943	6967	6972	6995	7000	7023
	7028	7052	7057	7082	7087	7166	7181	7184	7198	7201	7215	7218	7232	7235	7249
	7252	7266	7269	7283	7286	7300	7303	7317	7320	7334	7337	7375	7379	7384	7388
	7412	7416	7421	7424	7449	7453	7458	7461	7570	7586	7589	7606	7609	7627	7630
	7648	7651	7668	7671	7689	7692	7710	7713	7730	7733	7736	7758	7761	7764	7781
	7784	7787	7804	7807	7810	7827	7830	7833	7850	7853	7856	7873	7876	7879	7896
	7899	7902	7919	7922	7925	7942	7945	7948	7965	7968	7971	8000	8003	8006	8021
	8024	8027	8042	8045	8048	8064	8067	8070	8086	8089	8092	8107	8110	8113	8129
	8132	8135	8151	8154	8157	8172	8175	8178	8193	8196	8199	8215	8218	8221	8237
	8240	8243	8258	8261	8264	8279	8282	8285	8301	8304	8307	8323	8326	8329	8344
	8347	8350	8366	8369	8372	8392	8395	8398	8413	8416	8419	8434	8437	8440	8455
	8458	8461	8476	8479	8482	8497	8500	8503	8518	8521	8524	8539	8542	8545	8560
	8563	8566	8581	8584	8587	8609	8612	8615	8630	8633	8636	8652	8655	8658	8674
	8677	8680	8695	8698	8701	8717	8720	8723	8739	8742	8745	8760	8763	8766	8781
	8784	8787	8803	8806	8809	8825	8828	8831	8846	8849	8852	8868	8871	8874	8890
	8893	8896	8911	8914	8917	8934	8951	8970	8973	8976	8991	8994	8997	9012	9015
	9018	9033	9036	9039	9054	9057	9060	9075	9078	9081	9096	9099	9102	9117	9120
	9123	9138	9141	9144	9159	9162	9165	9178	9181	9292	9311	9331	9342	9358	9447
	9455	9466	9475	9493	9507	9514	9523	9539	9549	9573	9585	9615	9627	9652	9663

```
                 9727  9739  9750  9763  9825  9849  9873  9898  9919  9949  9973  9997 10021 10067 10099
                10127 10137 10146 10177 10187 10196 10219 10231 10275 10282 10290 10297 10333 10340 10348
                10355 10426 10436 10443 10451 10520 10530 10537 10545 10610 10638 10646 10669 10686 10695
                10723 10751 10767 10777 10784 10841 10906 10918 10929 10939 10950 10964 10976 10991 11016
                11033 11049 11063 11065 11080 11097 11113 11132 11170 11182 11193 11203 11214 11227 11239
                11253 11274 11284 11304 11311 11402 11416 11429 11443 11528 11556 11578 11588 11598 11607
                11617 11626 11635 11644 11672 11686 11691 11702 11729 11764 11767 11773 11779 11797 11800
                11806 11813 11882 11909 11923 11985 12012 12026 12413 12600 12684 12687 12710 12712 12717
                12733 12778 12784 12789 12841 12847 12852 12915 12924 12979 12986 12990 13042 13049 13110
                13116 13173 13177 13226 13231 13290 13308 13323 13338 13360 13365 13371 13424 13428 13445
                13449 13464 13468 13485 13489 13527 13532 13547 13552 13567 13572 13587 13592 13641 13645
                13707 13711 13770 13789 13811 13870 13874 13889 13893 13909 13913 13929 13933 13948 13952
                13967 13971 14046 14051 14066 14070 14087 14091 14107 14111 14126 14130 14145 14149 14218
                14223 14238 14243 14258 14263 14278 14282 14297 14301 14316 14321 14336 14341 14410 14415
                14430 14435 14485 14489 14505 14509 14574 14578 14598 14603 14621 14626 14641 14646 14661
                14666 14747 14752 14764 14768 14777 14782 14788 14805 14810 14824 14833 14838 14844 14861
                14866 14878 14882 14891 14896 14902 14966 14975 14980 14993 15004 15010 15023 15037 15055
                15079 15085 15100 15106 15375 15382 15425 15429 15446 15450 15464 15470 15478 15482 15658
                15661 15664 15815 15820 15991 15994 15997 16114 16119 16245 16250 16390 16395 16507 16511
                16525 16530 16533 16658 16662 16676 16680 16683 16890 16893 16897 16900 16903 17160 17168
                17172 17205 17228 17250 17312 17315 17319 17322 17325 17370 17399 17477 17485 17489 17521
                17528 17533 17537 17543 17547 17554 17558 17563 17568 17572 17577 17583 17587 17592 17599
                17606 17615 17623 17631 17640 17648 17696 17702 17712 17715 17752 17776 17813 17835 17853
                17863 17910 17985 18037 18088 18134 18185 18363 18539 18566 18603 18639 18665 18706 18773
                18821 18872 18923 18975 19021 19072 19115 19164 19212 19259 19307 19356 19407 19457 19504
                19550 19601 19654 19915 19956 19994 20027 20060 20092 20124 20158 20187 20214 20218 20466
                20643 20941 20985 21024 21063 21103 21144 21185 21227 21249 21278 21538 21686 21689 21693
                21732 21761 21764 21800 21816 21904 21918 21939 21945 21961 21974 21987 22001 22005 22009
                22014 22027 22031 22035 22041 22055 22059 22063 22069 22086 22091 22112 22119 22138 22147
                22169 22189 22196 22212 22233 22237 22259 22268 22289 22299 22323 22347 22365 22368 22390
                22417 22430 22434 22438 22448 22463 22470 22476 22496 22505 22513 22544 22552 22580 22601
                22641 22678 22692 22713 22719 22740 22753 22766 22780 22784 22788 22799 22812 22816 22820
                22830 22844 22848 22852 22862 22879 22884 22905 22912 22931 22940 22967 22985 22995 23012
                23030 23034 23055 23062 23081 23091 23113 23141 23157 23160 23186 23190 23206 23209 23224
                23228 23291 23361 23378 23396 23402 23407 23413
$$ERRO           500#  1003  1013  1023  1032  1057  1070  1084  1096  1110  1141  1150  1159  1168  1182
                1198  1212  1228  1243  1259  1274  1290  1306  1322  1337  1353  1378  1387  1396  1405
                1451  1481  1501  1525  1552  1584  1617  1650  1681  1717  1746  1771  1804  1842  1879
                1914  1942  1981  2006  2033  2055  2080  2100  2121  2144  2168  2191  2215  2249  2277
                2318  2347  2388  2414  2443  2470  2497  2515  2531  2553  2596  2622  2655  2680  2700
                2719  2739  2769  2790  2813  2837  2861  2879  2896  2913  2931  2949  2971  3004  3039
                3062  3093  3116  3138  3160  3184  3213  3246  3279  3306  3341  3378  3414  3448  3473
                3502  3530  3552  3581  3625  3673  3709  3733  3760  3781  3802  3824  3842  3864  3882
                3904  3928  3952  3976  4072  4189  4213  4247  4269  4290  4313  4357  4397  4421  4444
                4466  4520  4552  4597  4638  4652  4704  4746  4798  4838  4880  4925  4981  5005  5043
                5081  5118  5147  5188  5218  5234  5250  5265  5280  5295  5310  5336  5357  5378  5399
                5420  5441  5462  5485  5579  5662  5743  5761  5788  5816  5833  5843  5854  5864  5874
                5889  5905  5919  6076  6124  6158  6180  6269  6327  6340  6356  6395  6587  6659  6675
                6692  9341  9357  9522  9918 10274 10425 10519 10750 11062 17701 21960 22013 22211 22739
               22798 23011
$$ESCA           1#
$$NEWT           1#   500#  700   751   780   817   846   883   918   947   996  1006  1016  1026  1049
                1060  1073  1088  1099  1134  1144  1153  1162  1171  1186  1201  1216  1232  1247  1263
                1278  1295  1310  1326  1341  1371  1381  1390  1399  1438  1464  1491  1513  1536  1566
                1598  1631  1659  1690  1719  1748  1788  1820  1859  1882  1925  1961  1993  2019  2037
                2057  2082  2102  2132  2157  2180  2203  2226  2260  2288  2328  2359  2398  2424  2453
```

```
                2480    2506    2522    2538    2564    2604    2630    2662    2690    2709    2729    2753    2779    2800    2825
                2846    2871    2888    2905    2916    2934    2952    2974    3008    3042    3065    3096    3119    3141    3169
                3193    3221    3254    3289    3317    3353    3389    3424    3463    3492    3520    3542    3562    3594    3636
                3684    3721    3747    3772    3792    3815    3834    3853    3875    3893    3917    3944    3968    4014    4092
                4201    4229    4250    4272    4293    4328    4361    4411    4425    4447    4480    4524    4566    4601    4642
                4667    4708    4760    4801    4841    4883    4930    4985    5010    5054    5092    5128    5157    5201    5222
                5237    5253    5268    5283    5298    5321    5339    5360    5381    5402    5423    5444    5473    5494    5516
                5545    5563    5591    5633    5652    5679    5728    5746    5764    5791    5826    5836    5846    5857    5867
                5881    5896    5908    5922    6017    6078    6126    6162    6172    6183    6206    6216    6227    6238    6248
                6258    6271    6282    6303    6315    6329    6344    6360    6381    6398    6418    6444    6510    6529    6550
                6574    6590    6613    6646    6665    6683    6703    6736    6749    9281    9300    9319    9347    9377    9389
                9401    9413    9425    9437    9483    9500    9529    9563    9601    9638    9674    9709    9792    9927   10031
               10105   10155   10206   10252   10317   10399   10495   10590   10627   10705   10736   10757   10794   10861   10998
               11140   11260   11291   11504   11540   11565   11653   11710   11738   11851   11955   12060   12457   12655   12756
               12821   12883   12956   13017   13083   13150   13202   13261   13406   13509   13619   13686   13752   13852   14028
               14200   14392   14467   14554   14727   14953   15158   15402   15500   15679   15834   16012   16133   16266   16411
               16549   16700   16925   17191   17351   17513   17680   17727   17787   17871   17933   17993   18046   18096   18142
               18193   18373   18548   18575   18622   18647   18674   18730   18781   18828   18879   18930   18982   19028   19079
               19122   19171   19219   19266   19314   19364   19415   19464   19511   19557   19609   19662   19930   19966   20003
               20034   20067   20099   20131   20168   20197   20236   20480   20653   20956   20995   21035   21073   21113   21153
               21194   21235   21263   21292   21552   21576   21707   21748   21778   21895   21909   21922   21949   21965   21978
               21995   22017   22045   22074   22098   22123   22151   22177   22200   22216   22244   22273   22304   22328   22351
               22375   22408   22420   22451   22480   22524   22558   22584   22607   22669   22683   22696   22723   22744   22757
               22774   22802   22834   22867   22891   22916   22944   22975   23002   23016   23041   23067   23096   23118   23145
               23167   23195   23212   23261   23301

$$SKIP          1#
.EQUAT          1#
.HEADE          1#
.KT11           1#    9183#   9184
.SETUP          1#     543#   6896
.SWRHI          1#
.$ACTI          1#     543#
.$APTB          1#     543#    554
.$APTH          1#     543#    581
.$APTY          1#
.$ASTA          1#
.$CATC          1#
.$CMTA          1#
.$DB2D          1#
.$DB2O          1#
.$DIV           1#
.$EOP           1#
.$ERRO          1#
.$ERRT          1#
.$MULT          1#
.$POWE          1#
.$RAND          1#
.$RDDE          1#
.$RDOC          1#
.$READ          1#
.$R2AZ          1#
.$SAVE          1#
.$SB2D          1#
.$SB2O          1#
.$SCOP          1#
.$SIZE          1#
```

```
.$SUPR      1#
.$TRAP      1#
.$TYPB      1#
.$TYPD      1#
.$TYPE      1#
.$TYPO      1#
.$40CA      1#
.1170       1#


. ABS.  131275     000

 ERRORS DETECTED:  0

 DSKM:CJKDEA,DSKZ:CJKDEA.LST/SOL/CRF/NL:TOC=SYSMAC.SML,CJKDEA.P11
 RUN-TIME:  200 254 31 SECONDS
 RUN-TIME RATIO: 726/486=1.4
 CORE USED:  48K  (95 PAGES)
```